UNIVERSITY OF CALIFORNIA

Santa Barbara

On the Numerics, Generation, and Scaling of Fluvial Landscapes

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy

in

Mathematics

by

David James Cattan

Committee in charge:

Professor Björn Birnir, Chair

Professor Terry Smith

Professor Xu Yang

June 2017

This dissertation of David James Cattan is approved.

Professor Xu Yang

Professor Terry Smith

Professor Björn Birnir, Committee Chair

June 2017

On the Numerics, Generation, and Scaling of Fluvial Landscapes

David James Cattan

DEDICATION

To my wife, Jenny, and her patience with my "falling into degrees".

ACKNOWLEDGMENTS

We acknowledge support from the Center for Scientific Computing from the CNSI, MRL: an NSF MRSEC (DMR-1121053) and NSF CNS-0960316.

I also would like to thank the entirety of the UCSB Math Department for all of their support. I specifically would like to thank the Math Graduate Committee for approving my various petitions that led to this dissertation. I am especially indebted to Medina, the Graduate Advisor, for her help during my graduate years; especially with all those petitions.

I am thankful for my various employment opportunities with Collaborate. I have been very fortunate to work with, and for, some incredibly supportive people over the years.

Thank you to my friends and family that have been there for me over the years.

Most importantly, thank you to my committee: Professors Björn Birnir, Terry Smith, and Xu Yang. Thank you for all the patience, motivation, guidance, and general support you have given me over the years. As cliché as it might be, I truly had the best Ph.D. committee a graduate student could ask for.

Curriculum Vitæ

David James Cattan

EDUCATION

2011 - 2017	**PhD** Mathematics University of California, Santa Barbara
2011 - 2015	**MA** Applied Mathematics University of California, Santa Barbara
2007 - 2011	**BS** Mathematics **BS** Computer Science University of California, Santa Barbara

ACADEMIC EXPERIENCE

2014 - 2017	**Teaching Assistant** University of California, Santa Barbara
Summer 2016	**Teaching Associate** University of California, Santa Barbara
Spring 2014	**Course Grader** University of California, Santa Barbara
2013 - 2014	**Reviewer** International Journal of Nonlinear Sciences and Numerical Simulation
2009 - 2010	**Course Reader** University of California, Santa Barbara

NONACADEMIC EXPERIENCE

2012 - 2017	**Instructional Design Assistant** Collaborate Faculty Resource Center University of California, Santa Barbara
2010 - 2012	**Operations Coordinator** Collaborate Student Support Center University of California, Santa Barbara
2009 - 2010	**Student Consultant** Collaborate Student Resource Center University of California, Santa Barbara

Abstract

On the Numerics, Generation, and Scaling of Fluvial Landscapes

by

David James Cattan

The Smith and Bretherton model for fluvial erosion consists of a pair of partial differential equations: one governing water flow and one governing sediment flow. Numerical solutions of these equations have been shown to provide realistic models of the evolution of fluvial landscapes. Further analysis of these equations and their numerical solutions show that they possess scaling laws that are known to exist in nature. The preservation of these scaling laws in simulations is highly dependent on the numerical method used. Two numerical methods, both optimized for overland flow, have been used to simulate these surfaces. The implicit method exhibits the correct scaling laws, but the explicit method fails to do so. These equations, and the resulting models, help bridge the gap between the deterministic and stochastic theories of landscape evolution. Despite current advances in processing power and parallelism, numerical simulations of these surfaces take months of computation time. Some alterations can be made to code parameters to decrease computation time, but sacrifice accuracy of the resulting surfaces. Using the known deterministic and stochastic theories of these equations, the large scales of the model are generated from a series of elementary functions. The small scales are generated using Hurst fractal interpolation; a modified version of fractal interpolation functions, a relatively recent technique of interpolation explained herein. The generated surfaces provide great insight into the scaling laws satisfied by these surfaces.

Contents

Figures

Chapter 1

Introduction

Mathematical theories of erosion are generally measured by how well they model observable phenomena. The phenomena that one wants these models to capture include: (1) the emergence of channelized drainage patterns from unchanneled surfaces, (2) the development of relatively stable surfaces characterized by branching patterns of ridges and valleys, (3) the decline of the surfaces and the dissipation of the forms, and (4) the variability of landforms under varying environmental conditions. The theories must be based on physical principles and give rise to testable hypotheses. The groundbreaking work of Horton [28], combined with advances in our understanding of the evolution of fluvial landscapes, has caused models to be classified into three distinct approaches. These classifications are (i) deterministic modeling discrete in space and time based on conservation principles, (ii) stochastic modeling discrete in space and time based on conservation principles, and (iii) deterministic modeling based on the search for variational principles characterizing self-organizing drainage surfaces in terms of the minimization or maximization of an aggregate quantity.

Smith et al. [59, 62] developed a family of partial differential equations based on the conservation of water and sediment. These equations describe a transport-limited process [29] in which sediment moves in the same direction as the surface water flows. The transport-limited case models situations found in badlands and deserts where all the sediment can be transported away if a sufficient quantity of water is available.

The numerical implementation of a mathematical model plays a crucial role in validating the model. With a vast array of different numerical methods, choosing an appropriate method presents its own challenge. For the case of the Smith and Bretherton [60] fluvial erosion model, there is no choice between an implicit or explicit scheme. Small scale features crucial to the validity of this model are captured by implicit schemes, but not by explicit ones.

While the use of an appropriate numerical scheme is important, so is the time it takes to generate accurate results. Eroding a surface to its mature phase is of particular interest, as this is where many of the scaling results of these surfaces present themselves [12, 10]. Even with modern processing power and parallel computing, numerical simulations of the Smith and Bretherton model require months of computation time to reach this milestone.

The ability to quickly generate realistic looking landscapes has been a topic of great interest in recent years. Early techniques include random midpoint displacement [22], Poisson faulting [64], Fourier filtering [4, 54], and Perlin Noise [52]. Many of these techniques suffer from the same flaw of lack of feature control: the inability to determine where or even how many hills, mountains, ridges, valleys, etc. the resulting surface will contain. The work of Musgrave et al. [49] allows for some control over these features when using Perlin Noise to construct landscapes. Other work, such as that of Hnaidi et al. [27] and Génevaux et al. [24] for example, allow almost complete control as to the location of features in a synthetic landscape.

A solution in the middle of these extremes is discussed for the Smith and Bretherton model. A variety of methods to more quickly generate a surface in, or near, its mature phase that sacrifice as little accuracy as possible are presented. In particular, the known large scale dynamics and other results about these equations are used to select a series of elementary functions that, when combined, artificially construct solutions to the model. More recent techniques from fractal geometry are used for the necessary small scale dynamics.

The next chapter describes the family of partial differential equations in the model; the initial and boundary conditions, as well as the numerical implementations; and the necessary large and small scale features of numerical solutions. Chapter 3 discusses some of the modifications that have been made to the differential equations to improve realism in the resulting surfaces. The necessary background on fractal interpolation functions, how that method has been modified, and its application to fluvial landscapes is the subject of Chapter 4. Then Chapter 5 discusses methods of decreasing computation times, while still obtaining accurate results to the Smith and Bretherton model.

Chapter 2

The Numerics of Erosion

2.1 The Model of Erosion

Erosion is a highly non-linear process driven by noise, and thus difficult to represent mathematically. Below is a brief description and derivation of the model, its initial and boundary conditions, and the expected features of surfaces generated by this model. A more detailed derivation of the original model can be found in [59] and [12].

2.1.1 The Original Model

The basis of the model is the conservation of water and sediment over a continuous, erodible surface $z = z(x, y, t)$ and represents the advective entrainment and transport of sediment in transport-limited conditions [30]. The equations representing these processes

are

$$\frac{\partial h}{\partial t} = R + \nabla \cdot (\boldsymbol{u}_w q_w) \tag{2.1}$$

$$\frac{\partial z}{\partial t} = \nabla \cdot (\boldsymbol{u}_w q_s) \tag{2.2}$$

where $h = h(x, y, t)$ is the depth of water, which varies continuously over the land surface. The vector $-\boldsymbol{u}_w$ represents the direction of both the water and advected sediment flow, which is given by $-\boldsymbol{u}_w = -\nabla H/|\nabla H|$, where $H = H(x, y, t) = z(x, y, t) + h(x, y, t)$ is the height of the free water surface. The value R is a (constant) rainfall rate, and the fluxes of water and sediment per unit with are given by q_w and q_s, respectively.

A Manning-type relation is used for the flux of water,

$$q_w = nh^{5/3}|\nabla H|^{1/2} \tag{2.3}$$

where n is a constant representing the inverse of the channel roughness [15, 61, 63].

A power law relation is used for the sediment flux [40],

$$q_s = kh^{5\gamma/3}|\nabla H|^{\gamma/2+\delta} \tag{2.4}$$

in which k is an erosion coefficient, γ an δ are dimensionless erosion constants.

For ease of use, Eqs. (2.1) and (2.2) are transformed into a dimensionless form. For a variable v, it can be broken down as $v = [v]v^*$, where $[v]$ is a scale and v^* is the dimensionless variable. In this dimensionless form, there are three natural time scales that characterize the dynamics of these equations: a short, intermediate, and long time scale. More information about these time scales can be found in [12]. The model assumes the long time scale approximation, and dropping the asterisks from the dimensionless

variables yield

$$\eta^2 \frac{\partial h}{\partial t} = \nabla \cdot \left[\frac{\nabla H}{|\nabla H|} h^{\frac{5}{3}} |\nabla H|^{\frac{1}{2}} \right] + R \tag{2.5}$$

$$\frac{\partial H}{\partial t} - \eta \frac{\partial h}{\partial t} = \nabla \cdot \left[\frac{\nabla H}{|\nabla H|} h^{\frac{5}{3}\gamma} |\nabla H|^{\frac{\gamma}{2}+\delta} \right] \tag{2.6}$$

where $\eta \equiv [h]/[H] = [q_w]/[q_s]$ is a dimensionless "landscape" parameter.

The rate at which water flows down the surface is significantly shorter than the time it takes for even a small amount of sediment to erode. Thus, these two equations are solved over two different time scales. This difference in time scales leads to the assumption that the water depth h is at a statistically stationary state while the sediment erodes. Therefore, the $\eta \frac{\partial h}{\partial t}$ term is dropped from the left hand side of Eq. (2.6) in numerical calculations.

2.1.2 Initial and Boundary Conditions

Equations (2.5) and (2.6) are used to model a linear ridge extending uniformly in the lateral (x)-direction, defined over a rectangular domain of length L and width W,

$$\Omega = \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x \leq L,\, 0 \leq y \leq W\}.$$

The initial ridge is given by

$$H(x, y, 0) = cy + H_0 \tag{2.7}$$

where H_0 is some non-negative base height and c is the slope (grade) of the ridge.

The initial water depth satisfies

$$h(x, y, 0) = d(y), \quad d(0) = H_0, \quad d(W) = 0, \tag{2.8}$$

where $d(y)$ is a suitable continuous function. Except at the top of the ridge, water depths need to be positive to prevent instabilities from forming. Typically,

$$d(y) = H_0 \left(1 - \frac{y}{W}\right)$$

is used as the initial water depth.

The boundary conditions for the system are given by

$$h(x, W, t) = 0, \tag{2.9}$$

$$H(x, 0, t) = h(x, 0, t) = H_0. \tag{2.10}$$

Physically, these conditions state that there is no water at the top of the ridge and at the base of the ridge is some absorbing body of water; such as a lake or river. This body collects all the water and sediment that has run down the surface and carries it away.

The upper boundary also has a zero flux condition,

$$q_w = q_s = 0,$$

meaning that no water or sediment may pass through the upper boundary. The lateral boundaries at $x = 0$ and $x = L$ are periodic, modeling a ridge of infinite extent.

2.1.3 Features of the Model

The initial smooth surface $H(x, y, 0)$ given in the previous section is an unstable, steady-state solution of Eqs. (2.5) and (2.6) [11]. To obtain non-trivial results, the initial surface is seeded with a series of random perturbations. These perturbations break the symmetry of the surface, which allow for channel-forming instabilities to become effective, and ultimately drive the erosion process. These initially perturbed surfaces are actually unstable when eroded by water [12, 62]. After a characteristic period of erosion these unstable, perturbed surfaces evolve into statistically stationary landscapes [12, 10]. These statistically stationary landscapes, created from different arrangements of random perturbations, all contain the same statistics but the locations of the large scale features (described below) vary. Numerically, the instabilities of Eqs. (2.5) and (2.6) are saturated by these initial perturbations. This saturation transitions the model to a stochastic system. As a result, these surfaces have important deterministic and statistical scaling properties.

The deterministic results are referred to as "large scale" dynamics, and these are features that are clearly visible when observing these surfaces. These features include the number of mountain ridges, valleys, and rivers that form, along with the half-width of the valleys. It was found by Birnir et al. [10] that the half-width of the valleys is what determines these spacial scalings, but that the results can be written using properties of the model. The number of valleys N generated by this model in its mature phase is

$$N \sim \frac{L}{0.80(cW)^{4/3}}, \tag{2.11}$$

where L, W, and c are as defined in the previous section. For the typical domain size used in these models, it is expected that the mature surfaces will contain three mountain ridges separating three valleys, and that each of these valleys contains a primary river.

The statistical, or "small scale" feature of the landscape is its roughness formed by the turbulent flow of water over the surface [9]. This quantity is best captured by the variogram, which is given by

$$W_F(\boldsymbol{\ell}, t) = \langle |F(\boldsymbol{x} + \boldsymbol{\ell}, t) - F(\boldsymbol{x}, t)|^2 \rangle^{\frac{1}{2}}, \tag{2.12}$$

where $F(\boldsymbol{x}, t)$ is some function characterizing the surface, $\boldsymbol{\ell}$ is the lag variable, and $\langle \cdot \rangle$ represents the expectation operator. The expectation is approximated by taking the ensemble average of the desired quantity over multiple different surfaces.

It is not the variogram itself that is of importance, but its associated scaling exponent ζ [12] where

$$W_F(\boldsymbol{\ell}, t) \sim |\boldsymbol{\ell}|^{\zeta}, \tag{2.13}$$

which is calculated by finding the slope of the log-log plot. The exponent ζ is the Hurst exponent [42, 43, 44] and its significance is discussed more in Sect. 2.3.

The variogram (also known as a height-height correlation function, roughness function, or width function) measures the correlation in values of $F(\boldsymbol{x}, t)$ at points separated by a distance $|\boldsymbol{\ell}|$, which is a measure of the roughness of the function F [12]. For these surfaces the water depth h, water surface height H, and the slope of the water surface $|\nabla H|$ are all used as the characterizing function.

Due to the initial random perturbations and other instabilities of the surface, the small scale features that develop are stochastic in nature. The variogram is the primary tool to quantify these statistical quantities in a deterministic way [10]. These small scales are also used as a measure of accuracy for this model as typical stability tests do not apply to this non-linear model.

2.2 Numeric Implementation

Equations (2.5) and (2.6) were solved for numerically using two different schemes: a MacCormack predictor-corrector method and an upwind/Crank-Nicolson method. Both these schemes are optimized for two-dimensional overland flow, but one is explicit while the other is implicit. Comparing the resulting surfaces from these methods has shown that the large scale dynamics discussed in Sect. 2.1.3 are captured by both methods, but only one method also captures the small scales.

Both schemes were solved on a domain of 100m by 100m which was divided into square grid cells of 1m by 1m. The initial slope was a 5% grade, and had grid cells with random perturbations up to 1 cm in height. The constants δ and γ were both set to 2.

The general procedure for solving this system begins with calculating the water depth over the surface. When solving Eq. (2.5) numerically, multiple iterations are performed until an equilibrium water depth emerges. Once the equilibrium water depth has been found, the surface is eroded by a small amount; no more than one millimeter of sediment can be removed from a single location in order to comply with the CFL condition and maintain stability [62]. This process repeats until the desired amount of sediment has been eroded from the initial surface.

Using both schemes, the initial linear ridges were eroded through their channelization phases and into their mature phase, which contain the features of interest discussed in Sect. 2.1.3. The work of Simpson and Schlunegger [57] uses the same base of the Smith and Bretherton [60] model solved with a finite element method on a triangular mesh. This discretization has the primary disadvantage of greater computational complexity, but an advantage of water and sediment moving in a more realistic manner. Simpson and Castelltort [56] used a more physically based model of five coupled differential equa-

tions for water flow and sediment transport. Both models were used to simulate the channelization of a linear ridge, but nothing further in the process. These channelization simulations showed similar results to those of this model.

Other work in fluvial systems uses similar numerical techniques as those described below. For example, Fowler et al. [23] used a second-order upwind scheme along with a Euler method using second-order finite differences in their work in channel formation. Another is the work of Perron et al. [53], who solved a single conservation of mass equation in their work on the spacing of river valleys. This equation was solved using a splitting method; one fractional step was solved using a second-order Runge-Kutta scheme and the second was solved via a Crank-Nicolson scheme.

2.2.1 The Predictor-Corrector Scheme

Smith et al. [59, 62] numerically solved Eqs. (2.5) and (2.6) by implementing the explicit MacCormack two-step predictor corrector method for two dimensional overland flow [69]. As the name implies, this scheme consists of two separate predictor-corrector procedures. Starting from time t_n, a solution is obtained at time $t_{n+\frac{1}{2}}$, which involves a backward difference predictor step, followed by a forward difference corrector step. This data is then used to solve for the data at time t_{n+1} by using a forward difference predictor and then a backward difference corrector step.

The MacCormack scheme has been verified for its realistic simulations of two-dimensional flow and can be used to solve for both the water and sediment flows [39]. Let P represent either the water depth h or the height of the water surface H. Let U and V be the flow of the quantity being solved for in the x and y directions, respectively. Then the difference equations for this scheme are as follows:

The time $t_{n+\frac{1}{2}}$ predictor step (backward difference)

$$\overline{P}_{ij}^{n+\frac{1}{2}} = P_{ij}^{n} - \frac{\Delta t}{2\Delta x}\left(U_{ij}^{n} - U_{i-1\,j}^{n}\right) - \frac{\Delta t}{2\Delta y}\left(V_{ij}^{n} - V_{ij-1}^{n}\right) + \frac{R\Delta t}{2}.$$

The time $t_{n+\frac{1}{2}}$ corrector step (forward difference)

$$P_{ij}^{n+\frac{1}{2}} = \frac{1}{2}\left[P_{ij}^{n} + \overline{P}_{ij}^{n+\frac{1}{2}} - \frac{\Delta t}{2\Delta x}\left(\overline{U}_{i+1\,j}^{n+\frac{1}{2}} - \overline{U}_{ij}^{n+\frac{1}{2}}\right) - \frac{\Delta t}{2\Delta y}\left(\overline{V}_{ij+1}^{n+\frac{1}{2}} - \overline{V}_{ij}^{n+\frac{1}{2}}\right) + \frac{R\Delta t}{2}\right].$$

The time t_{n+1} predictor step (forward difference)

$$\overline{P}_{ij}^{n+1} = P_{ij}^{n+\frac{1}{2}} - \frac{\Delta t}{2\Delta x}\left(U_{i+1\,j}^{n+\frac{1}{2}} - U_{ij}^{n+\frac{1}{2}}\right) - \frac{\Delta t}{2\Delta y}\left(V_{ij+1}^{n+\frac{1}{2}} - V_{ij}^{n+\frac{1}{2}}\right) + \frac{R\Delta t}{2}.$$

The time t_{n+1} corrector step (backward difference)

$$P_{ij}^{n} = \frac{1}{2}\left[P_{ij}^{n+\frac{1}{2}} + \overline{P}_{ij}^{n+1} - \frac{\Delta t}{2\Delta x}\left(\overline{U}_{ij}^{n+1} - \overline{U}_{i-1\,j}^{n+1}\right) - \frac{\Delta t}{2\Delta y}\left(\overline{V}_{ij}^{n+1} - \overline{V}_{ij-1}^{n+1}\right) + \frac{R\Delta t}{2}\right].$$

The term R is the rainfall rate from Eq. (2.5) and is set to zero when solving for the sediment flow. To simplify calculations, square grid cells are assumed and the landscape parameter η is set to 1. After each step, the flow variables U and V need to be recalculated. Like with the variable P, the bar signifies values that are calculated after a predictor step, and the superscript indicates which time step that value was calculated.

Artificial Viscosity

When solving for the water flow, oscillations in depth cause small numerical instabilities in this scheme. These instabilities can amplify and ruin the calculations, resulting in

unstable and unrealistic surfaces. In order to prevent this, a small amount of artificial viscosity must be added to the scheme after each corrector step [19]. The amount of viscosity to add to grid location (i, j) is given by

$$V_x h_{i\,j} + V_y h_{i\,j} \tag{2.14}$$

where V_x and V_y are operators in the x and y directions, respectively. These operators are defined by

$$V_x h_{i\,j} = \varepsilon_{i+\frac{1}{2}\,j}(h_{i+1\,j} - h_{i\,j}) - \varepsilon_{i-\frac{1}{2}\,j}(h_{i\,j} - h_{i-1\,j}), \tag{2.15}$$

$$V_y h_{i\,j} = \varepsilon_{i\,j+\frac{1}{2}}(h_{i\,j+1} - h_{i\,j}) - \varepsilon_{i\,j-\frac{1}{2}}(h_{i\,j} - h_{i\,j-1}), \tag{2.16}$$

where

$$\begin{aligned}
\varepsilon_{i-\frac{1}{2}\,j} &= \max(W_{i-1\,j}, W_{i\,j}) \cdot K, \\
\varepsilon_{i+\frac{1}{2}\,j} &= \max(W_{i+1\,j}, W_{i\,j}) \cdot K, \\
\varepsilon_{i\,j-\frac{1}{2}} &= \max(X_{i\,j-1}, X_{i\,j}) \cdot K, \\
\varepsilon_{i\,j+\frac{1}{2}} &= \max(X_{i\,j+1}, X_{i\,j}) \cdot K,
\end{aligned}$$

$$W_{i\,j} = \frac{|h_{i+1\,j} - 2h_{i\,j} + h_{i-1\,j}|}{|h_{i+1\,j}| + 2|h_{i\,j}| + |h_{i-1\,j}|},$$

$$X_{i\,j} = \frac{|h_{i\,j+1} - 2h_{i\,j} + h_{i\,j-1}|}{|h_{i\,j+1}| + 2|h_{i\,j}| + |h_{i\,j-1}|}.$$

The difficulty in adding the artificial viscosity is the parameter K, which controls the amount of viscosity added, and typically has a value between 0.5 and 3 [19]. Unfortunately, there is no formula or algorithm to determine its value. If K is set too small it has no effect, water depths can become negative, and the method becomes unstable. If K is

too large it over dampens the system and can hinder the erosion process, possibly causing a geomorphologically concave landscape with no discernible features. For an appropriate amount of viscosity, this scheme is second order in both space and time [62].

2.2.2 The Upwind Scheme

A standard method for hyperbolic equations, the upwind scheme is an alternative method to solve for the water flow. One benefit of this scheme is that it does not require the use of artificial viscosity. This method preserves shocks that form in the water flow in a controlled manner [55].

The flux of water can be written as $q_w = hV_w$ where $V_w = u\hat{x} + v\hat{y}$ is the water flow velocity per unit width; u, v are the water flow velocities; and $\hat{x}$, $\hat{y}$ are unit vectors in the x and y directions, respectively.

Take Eq. (2.1) and substitute in this new representation of the water flux,

$$\frac{\partial h}{\partial t} + \nabla \cdot (hu\hat{x} + hv\hat{y}) = R. \tag{2.17}$$

Expanding the differential operator in Eq. (2.5) yields

$$\frac{\partial h}{\partial t} = \frac{\partial}{\partial x}\left(\frac{nH_x h^{5/3}}{|\nabla H|^{\frac{1}{2}}}\right) + \frac{\partial}{\partial y}\left(\frac{nH_y h^{5/3}}{|\nabla H|^{\frac{1}{2}}}\right) + R, \tag{2.18}$$

where H_x and H_y are the partial derivatives of H in x and y, respectively. Comparing these two equations yields that

$$u = -\frac{nH_x h^{\frac{2}{3}}}{|\nabla H|^{\frac{1}{2}}} \qquad \text{and} \qquad v = -\frac{nH_y h^{\frac{2}{3}}}{|\nabla H|^{\frac{1}{2}}}.$$

Then Eq. (2.18) can be written as

$$\frac{\partial h}{\partial t} = -\frac{\partial}{\partial x}(hu) - \frac{\partial}{\partial y}(hv) + R. \tag{2.19}$$

Discretizing Eq. (2.19) over half-grid cells yields

$$\frac{h_{ij}^{n+1} - h_{ij}^{n}}{\Delta t} = -\frac{\left(h_{i+*\,j}^{n} u_{i+\frac{1}{2}\,j}^{n} - h_{i-*\,j}^{n} u_{i-\frac{1}{2}\,j}^{n}\right)}{\Delta x} - \frac{\left(h_{i\,j+*}^{n} v_{i\,j+\frac{1}{2}}^{n} - h_{i\,j-*}^{n} v_{i\,j-\frac{1}{2}}^{n}\right)}{\Delta y} + R \tag{2.20}$$

where $u_{i\pm\frac{1}{2}\,j}^{n}$ and $v_{i\,j\pm\frac{1}{2}}^{n}$ are solved using the appropriate forward or backward difference formula for H_x or H_y, respectively. The terms $h_{i\pm*\,j}^{n}$ and $h_{i\,j\pm*}^{n}$ represent the water depth at an initially unknown point. The grid cell from which the water depth is taken depends on the sign(s) of u and v. The sign of these variables determines the "downhill" direction, and thus the direction in which water is moving at a particular point.

The left-hand side of Eq. (2.20) is centered at time $t_{n+\frac{1}{2}}$ while the right-hand side is centered at time t_n, causing an error of first order in time. Similarly, the left-hand side is spacially centered at grid point (i, j) while the right-hand side is not. Therefore, this discretization is first order accurate in both space and time.

2.2.3 The Crank-Nicolson Scheme

Equation (2.6) is solved using a Crank-Nicolson scheme which is second order accurate in both space and time. The Crank-Nicolson scheme is an implicit method that takes the average value of the discretizations at times t_n and t_{n+1}. To discretize, write Eq. (2.6) as

$$\frac{\partial H}{\partial t} = \nabla \cdot \left(\nabla H \cdot h^{B} |\nabla H|^{C}\right) \tag{2.21}$$

where $B = \frac{5}{3}\gamma$ and $C = \frac{\gamma}{2} + \delta - 1$. This expands to

$$\frac{\partial H}{\partial t} = \frac{\partial}{\partial x}\left[h^B H_x \left(H_x^2 + H_y^2\right)^{C/2}\right] + \frac{\partial}{\partial y}\left[h^B H_y \left(H_x^2 + H_y^2\right)^{C/2}\right]. \tag{2.22}$$

Let $D_{i\,j}$ be a value to be differentiated. Define the operators

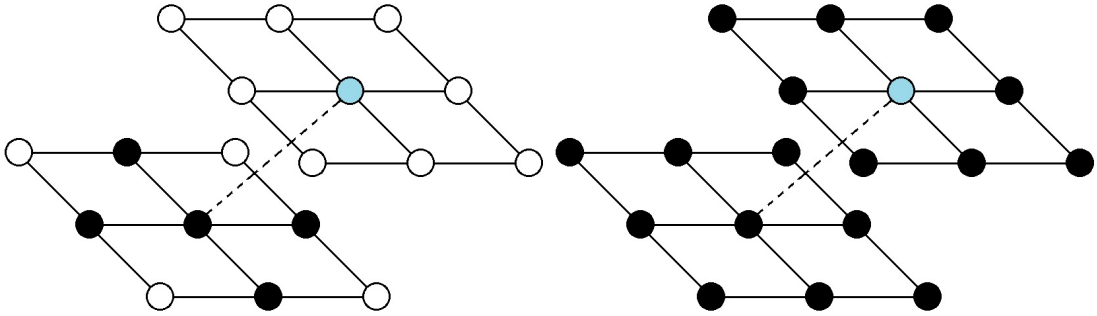
$$\delta_x(D_{i\,j}) = \frac{D_{i+\frac{1}{2}\,j} - D_{i-\frac{1}{2}\,j}}{\Delta x} \quad \text{and} \quad \delta_y(D_{i\,j}) = \frac{D_{i\,j+\frac{1}{2}} - D_{i\,j-\frac{1}{2}}}{\Delta y}$$

to be the approximations of the x and y derivatives, respectively. The Crank-Nicolson scheme with these derivative approximations becomes

$$\begin{aligned}\frac{H_{ij}^{n+1} - H_{ij}^n}{\Delta t} = {} & \frac{E}{2}\delta_x\left[(h_{i\,j})^B H_{i\,j,x}^{n+1}\left[\left(H_{i\,j,x}^{n+1}\right)^2 + \left(H_{i\,j,y}^{n+1}\right)^2\right]^{C/2}\right] \\ & + \frac{E}{2}\delta_x\left[(h_{i\,j})^B H_{i\,j,x}^{n}\left[\left(H_{i\,j,x}^{n}\right)^2 + \left(H_{i\,j,y}^{n}\right)^2\right]^{C/2}\right] \\ & + \frac{E}{2}\delta_y\left[(h_{i\,j})^B H_{i\,j,y}^{n+1}\left[\left(H_{i\,j,x}^{n+1}\right)^2 + \left(H_{i\,j,y}^{n+1}\right)^2\right]^{C/2}\right] \\ & + \frac{E}{2}\delta_y\left[(h_{i\,j})^B H_{i\,j,y}^{n}\left[\left(H_{i\,j,x}^{n}\right)^2 + \left(H_{i\,j,y}^{n}\right)^2\right]^{C/2}\right],\end{aligned} \tag{2.23}$$

where E is an erosion coefficient. In order to implement Eq. (2.23) numerically, it must be linearized about the values of H from the future time t_{n+1}. Once the equation is linear in time t_{n+1}, all the partial derivatives are expressed using appropriate numerical approximations.

The stencil for solving the resulting equation is given in Fig. 2.1b. Solving for the value H_{ij}^{n+1} requires the surrounding values of H^{n+1}, which can make this equation difficult to solve. Smith et al. [62] originally solved this equation using a successive over-relaxation scheme. Currently, this equation is solved by using a factoring scheme to create four tridiagonal systems, each of which is solved using the Thomas algorithm.



(a) Stencil for Upwind and MacCormack (b) Stencil for Crank-Nicolson

Figure 2.1: The two different stencils used in the discretization of this problem. The blue node is the value being solved for, the black nodes are required neighboring data, and the white nodes are neighboring data that are not required for the calculation.

The Factoring Scheme

The factoring scheme takes a multi-dimensional problem and transforms it into a series of single dimensional problems, which can then be solved for in parallel.

As an example, consider the heat equation

$$\frac{\partial \Phi}{\partial t} = k\nabla^2\Phi,$$

which is a simpler version of Eq. (2.6), especially for demonstration purposes. Using the Crank-Nicolson scheme, this can be written as

$$\frac{\Phi^{n+1} - \Phi^n}{\Delta t} = \frac{k}{2}\nabla^2\Phi^{n+1} + \frac{k}{2}\nabla^2\Phi^n.$$

Let A_x and A_y be the linear difference operators used to approximate the second derivative in x and y, respectively. Then the equation discretizes to

$$\frac{\Phi^{n+1} - \Phi^n}{\Delta t} = \frac{k}{2}A_x(\Phi^{n+1} + \Phi^n) + \frac{k}{2}A_y(\Phi^{n+1} + \Phi^n).$$

Let $\alpha = (k\Delta t)/2$. Grouping like time steps in Φ yields

$$\Phi^{n+1} - \alpha A_x(\Phi^{n+1}) - \alpha A_y(\Phi^{n+1}) = \Phi^n + \alpha A_x(\Phi^n) + \alpha A_y(\Phi^n).$$

Finally, the result gets "factored" to

$$(I - \alpha A_x)(I - \alpha A_y)\Phi^{n+1} = (I + \alpha A_x)(I + \alpha A_y)\Phi^n$$

where I is the identity matrix. This creates two nested, one-dimensional systems, which are all independent [48]. The difference in the cross terms is of order $(\Delta t)^3$, and thus does not affect the accuracy of the method.

The heat equation results in a five-point stencil (similar to Fig. 2.1a), which when using the factoring scheme yields two, one-dimensional systems as shown above. The discretization of Eq. (2.23) results in a nine-point stencil, requiring all eight surrounding points to calculate a value. Thus the factoring scheme yields four, one-dimensional systems instead of just two: the x, y directions and the two "diagonal" directions. The full discretization of Eq. (2.6) using the Crank-Nicoloson scheme and the factoring scheme can be found in Appendix A.

2.3 Comparisons

The surface on the left in Fig. 2.2 was generated using the implicit upwind/Crank-Nicolson scheme, while the surface on the right was generated using the explicit MacCormak method. As discussed in Sect. 2.1.3, both landscapes have developed the expected three primary mountain ridges separating three river valleys. Each surface also contains smaller, unstable collapsing mountain ridges still in the process of eroding away. Thus

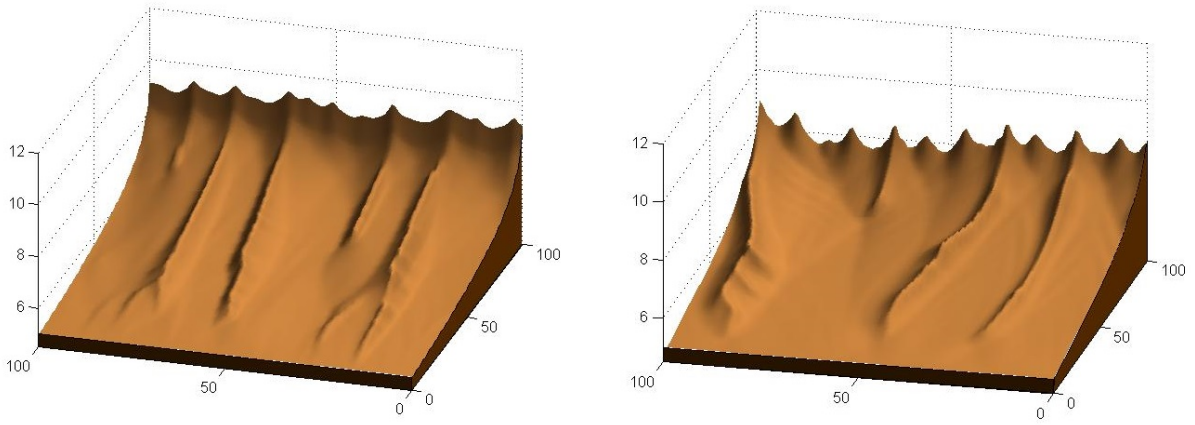


Figure 2.2: Typical surfaces after 60% of sediment has been eroded.

both surfaces contain the large scale dynamics expected from the theory.

The small scale dynamics tell a different story. The variogram, given in Eq. (2.12), is calculated using an ensemble average over multiple surfaces. As mentioned in Sect. 2.1.3, the variogram is calculated using one of three different variables characterizing the landscape, and it is expected that scaling results be different for each of these values. It was found by Birnir et al. [12] that surfaces generated by the MacCormack scheme produced the same scaling exponent for each variable; and that exponent depended on the artificial viscosity. However, surfaces generated by the upwind/Crank-Nicolson scheme produced different scaling exponents.

The most accurate readings for the variogram arise in the center of the surface as the imposed boundary conditions skew the scaling [10]. The variogram in H, shown in Fig. 2.3, for the implicit scheme reaches 0.75 for the scaling exponent, with an average value of about 0.70. The scaling exponent for the gradient of the water surface, $|\nabla H|$, achieves a scaling exponent value of 0.30 (Fig. 2.4) with the average at about 0.25.

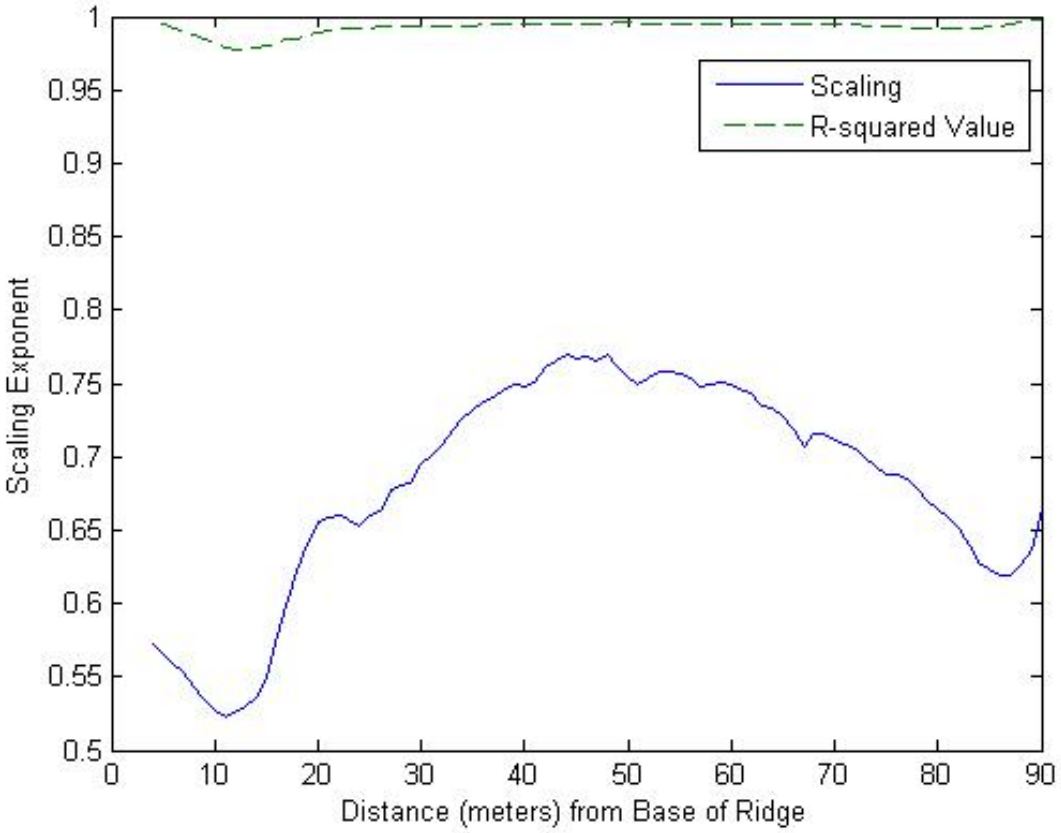


Figure 2.3: Variogram of H.

These scaling exponents are significant not only because they differ, but because of Hack's Law [25, 9]. If a river has length ℓ and it runs through a river basin of area A, Hack's Law states that $\ell \sim A^{0.58}$. The river basin has some average width, perpendicular to the river. This width can be represented by ℓ^{χ} where χ is the roughness exponent (and $1+\chi$ is the avalanche dimension). Then the area of the river basin is $A \sim \ell^{1+\chi}$. In order to be consistent with Hack's Law,

$$\frac{1}{1+\chi} = 0.58,$$

and thus the roughness exponent $\chi \approx 3/4$.

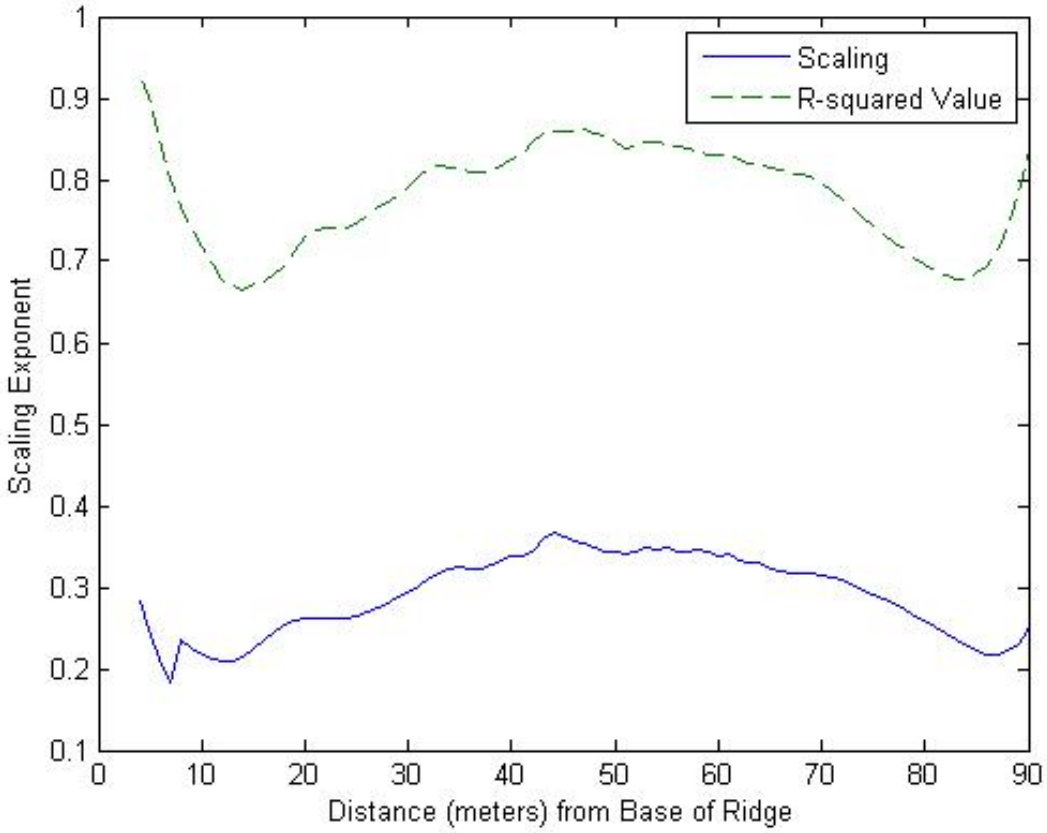


Figure 2.4: Variogram of $|\nabla H|$.

The scaling (Hurst) exponent computed from the variogram is related to the roughness exponent. If $\boldsymbol{\ell}$ is the lag variable, as in Eq. (2.12), then

$$\langle |F(\boldsymbol{\ell} + \boldsymbol{x}, t) - F(\boldsymbol{x}, t)|^2 \rangle^{\frac{1}{2}} \sim |\boldsymbol{\ell}|^{1-\chi} \tag{2.24}$$

where χ is the roughness exponent from above and $1-\chi$ is the Hurst exponent [42, 43, 44]. From Hack's Law, it is expected that the scaling (Hurst exponent) for the water surface H should be around 0.25 instead. The Hurst exponent is a measure of smoothness, and the general topography smooths the water surface H, thus increasing the scaling of the variogram. The scaling exponent for water surface H is contaminated by the large scales,

whereas the gradient $|\nabla H|$ gives the correct scaling exponent. This contamination by the large scales explains why the (Hurst) scaling exponent for the water surface H is larger than it should be. Thus it is actually the gradient $|\nabla H|$ that produces the correct scaling, and not the water surface H as previously thought [12].

This result of a larger variogram scaling (Hurst exponent) is confirmed by other findings. One example is the work of Simpson and Schlunegger [57], who found the scaling exponents of their simulated surfaces fell in the range 0.70 - 0.80. Another is the work of Voss [65] who found that mountainous landscapes have a Hurst exponent near 0.80 in value. Field values of these scalings (Hurt exponents) for landscapes in Ethiopia, Somalia, and Saudi Arabia were found to fall in the range of 0.50 - 0.75 [66]. This last result is of particular interest as these areas resemble the badland conditions simulated by the Smith and Bretherton model.

The roughness coefficient for the gradient of the water surface, $|\nabla H|$, was computed to be $\chi = 0.70$ by Birnir et al. [12]. This scaling more closely matches the current numerically computed scaling $\chi = 0.75$. The reason for this is that the gradient measures a local quantity via the difference quotient, thus the general shape of the landscape does not skew the results as it does for the water surface.

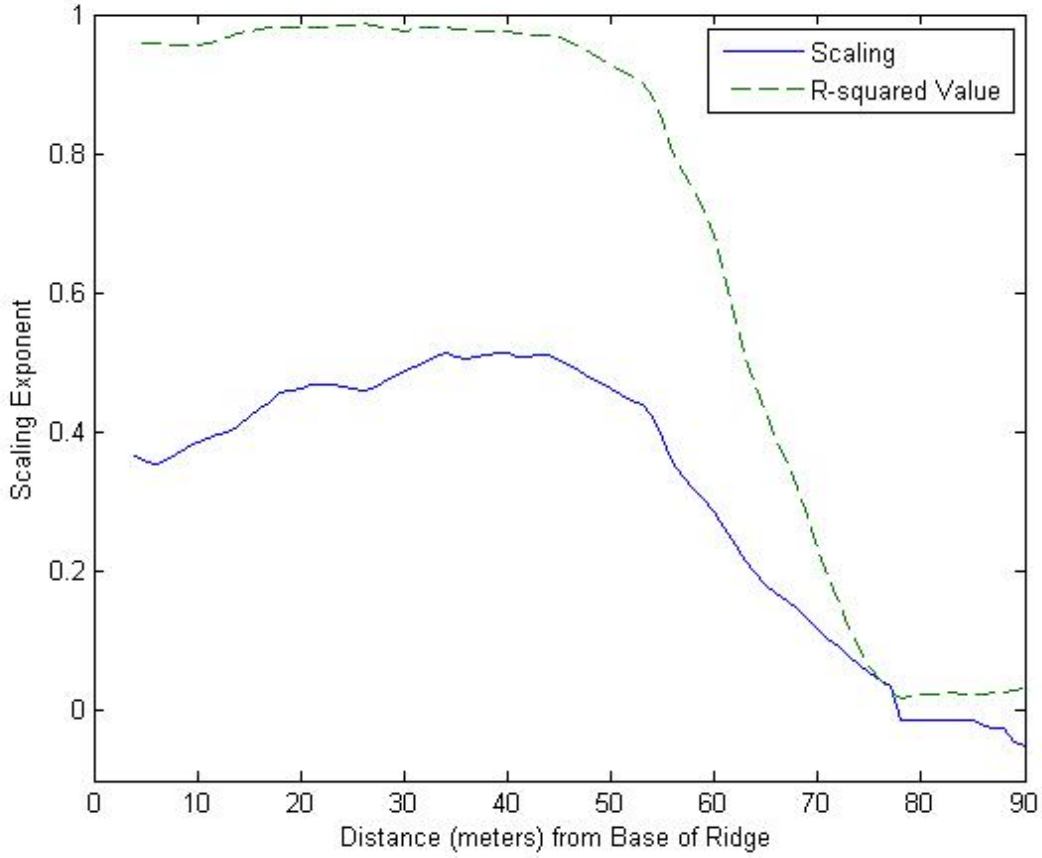


Figure 2.5: Variogram of h.

The scaling exponents for H and $|\nabla H|$ are both calculated for surfaces around when 60% of the initial sediment has been eroded. The scaling of the water depth h is instead calculated after 10% of sediment has been eroded, and is calculated to be 0.50, which is the scaling of Brownian motion [42, 43]. This scaling corresponds to the movement of water over the surface in the channelization process (see [17]). As the surface evolves into a mature landscape, turbulence in the overland flow ruins the scaling caused by the initial Brownian motion, which is why the scaling must be calculated early in the process.

Chapter 3

Model Modifications

While the Smith and Bretherton model does capture the necessary scaling properties of fluvial surfaces (see Sect. 2.3), there are still features this model could simulate more realistically. One goal is to modify this model to better capture the formation of river channels; which are defined here as concentrations of water flow (and sediment transport) within well-defined banks [8]. While simulations of Eqs. (2.5) and (2.6) do allow for the formation of rivers, they are not a noticeably dominant feature as the mountain ridges or valleys.

Figure 3.1 is a typical surface generated by this model after 60% of the initial sediment has been eroded. The deterministic features mentioned previously are clearly visible: there are three primary mountain ridges separating three river valleys. The collapsing ridge(s) are a typical feature of the model, but not as dominate or predictable as the primary (longer) ridges. If rivers are present in this model, one would expect noticeable evidence of their existence, e.g., river banks. There is some evidence of these formations at the base of the mountain ridges, however it is inconclusive as to whether these are actually rills or banks.

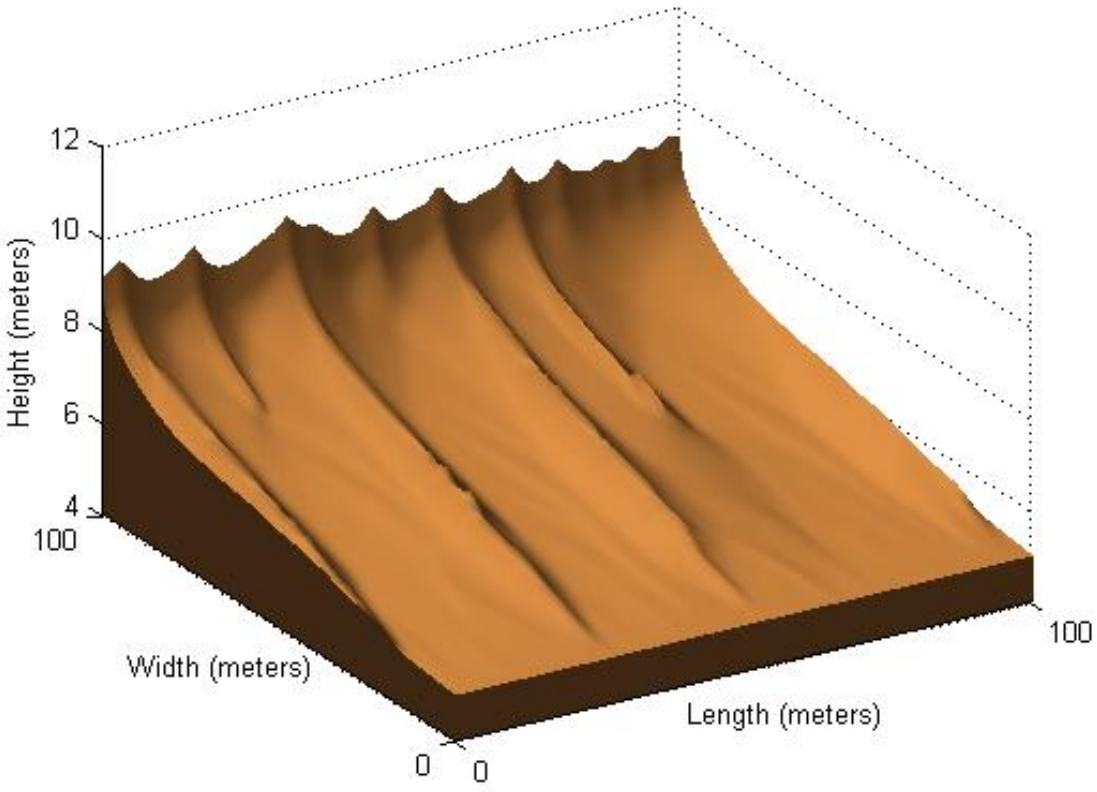


Figure 3.1: A typical surface at 60% eroded.

Currently, to determine the location of the rivers within the model, it is required to look at a graph of the water depth (see Fig. 3.2). From this data, it is apparent that a primary river that has formed in each of the three valleys. As mentioned above, a major goal is to modify this model to allow these rivers to dig themselves down into the surface. The hope is that the digging of rivers into the surface will make their banks more pronounced, so that they are visible when viewing a plot of the water surface. There are two modification attempts of Eq. (2.6) to force this extra digging, discussed below.

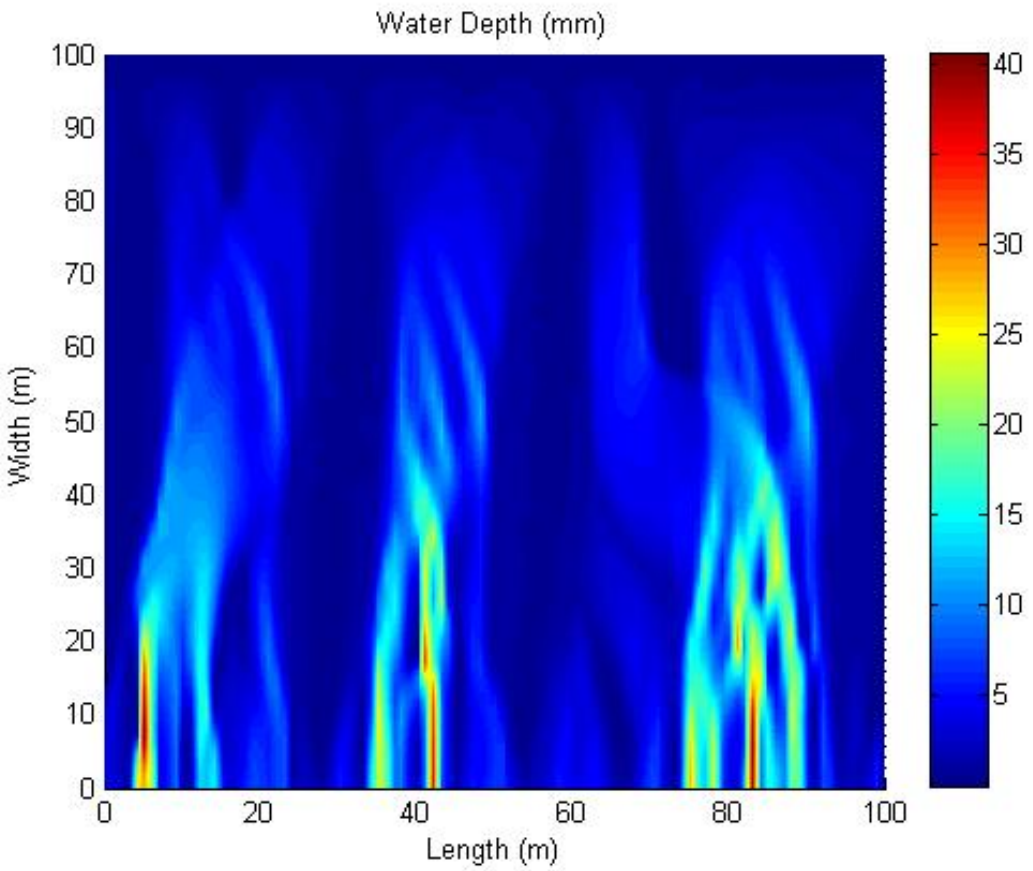


Figure 3.2: The depth of water h over the surface in Figure 3.1.

3.1 The Diffusion Term

In the early stages of erosion, the original model generates a series of braided rivers, defined by multiple concentrations of water intersecting one-another in a braided fashion. As the surface erodes, these braided rivers slowly transition into meandering rivers. The river on the right side of Fig. 3.2 is still evident of its original braided nature. The river on the left is primarily a meandering river: with one primary thalweg or multiple channels that do not intersect as with a braided river. The inspiration for this first modification was to force the transition of braided to meandering rivers to occur earlier in the erosion process, thus allowing more time for the meandering rivers to dig out their channels.

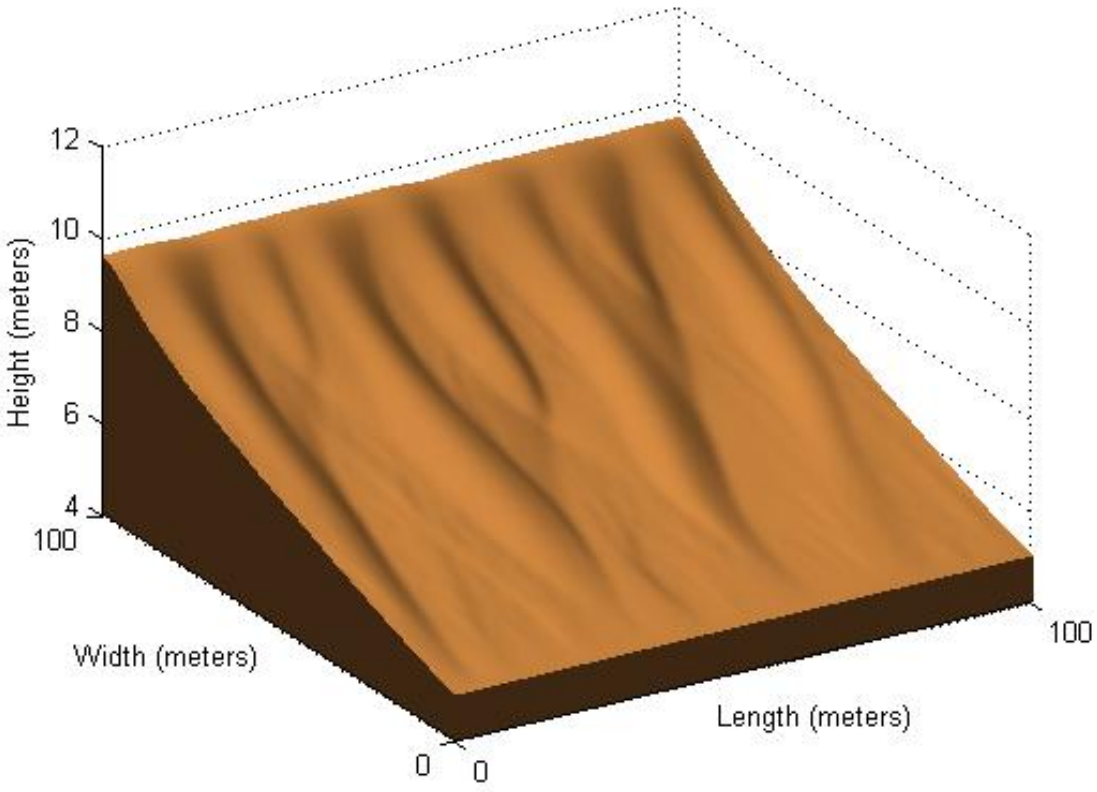


Figure 3.3: A surface with diffusion.

One factor that causes rivers to meander is the movement of sediment from the outer river bank to its inner bank [33]. The goal was to add a term to simulate this movement of sediment; the natural choice being the diffusion operator. The modified sediment flow equation is given by

$$\frac{\partial H}{\partial t} = \nabla \cdot \left[\frac{\nabla H}{|\nabla H|} h^{\frac{5}{3}\gamma} |\nabla H|^{\frac{\gamma}{2}+\delta} \right] + \beta \nabla^2 H + \mathcal{U}, \tag{3.1}$$

where β is a diffusion coefficient and $\mathcal{U}$ is a constant tectonic uplift rate. The uplift term is added both for realism and the idea that the uplift of the land surface paired with the digging of the river beds will create more pronounced channels.

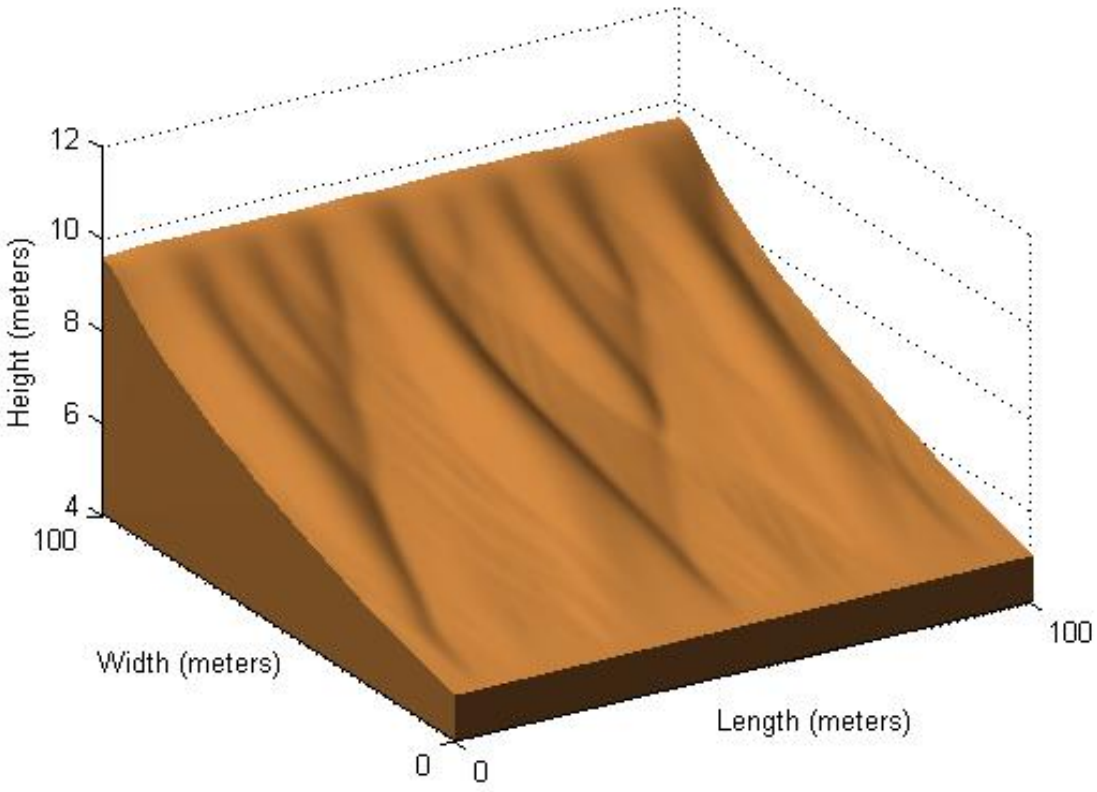


Figure 3.4: A surface with diffusion down the z gradient.

The addition of this diffusion term was a partial success. Rivers formed by this model meandered more freely and dug clearer channels into the surface, see Fig. 3.3. Instead of just altering the rivers, the dissipation term altered the entire surface from the expected results; creating a series of dissipative (or rolling) hills.

Smith [58] discusses the difference between movement of sediment down the water surface H and the land surface z. Based on this distinction the modification

$$\frac{\partial H}{\partial t} = \nabla \cdot \left[\frac{\nabla H}{|\nabla H|} h^{\frac{5}{3}\gamma} |\nabla H|^{\frac{\gamma}{2}+\delta} \right] + \beta \nabla^2 z + \mathcal{U}, \tag{3.2}$$

was made to the model. This addition had very similar results to Eq. (3.1) as can be seen

in Fig. 3.4. The primary difference is where channels dug themselves down. Figures 3.3 and 3.4 show a clear distinction between these two directions of flow.

The surfaces generated by these two diffusion terms no longer possess the large scale dynamics found by Birnir et al. [12, 10]. The expected scaling exponent for surfaces generated by this model is near 0.75 (see Sect. 2.3). As diffusive surfaces, it is expected that the scaling exponent of these surfaces is closer to 0.5 [12]. Due to the long computation times of the model (see Chapter 5), confirmation of this expected result, along with any large scale features expected from these altered models, are to be determined by future work.

3.2 The Abrasion Term

Shocks, eddies, and other forms of turbulence are a driving force in rivers digging out their channels [9, 34, 67]. It was Kramer and Marder [41] that inspired the following abrasion term to simulate the movement of sediment along the bottom of a river caused by turbulent flow. The sediment flow equation with this term is given by

$$\frac{\partial H}{\partial t} = \nabla \cdot \left[\frac{\nabla H}{|\nabla H|} h^{\frac{5}{3}\gamma} |\nabla H|^{\frac{\gamma}{2}+\delta} \right] - \alpha h^{3/2} |\nabla H| + \mathcal{U} \tag{3.3}$$

where α is an abrasion constant. The uplift term $\mathcal{U}$ is added for the same reasons as with the diffusion term(s). A similar abrasion term is proposed by Smith [58] and is expected to give analogous results.

Initial tests with the abrasion term proved successful, although more are required. Figure 3.5 shows one of the surfaces generated using this abrasion term. Upon close inspection, there is evidence of what could be river banks or rills within the valleys. The

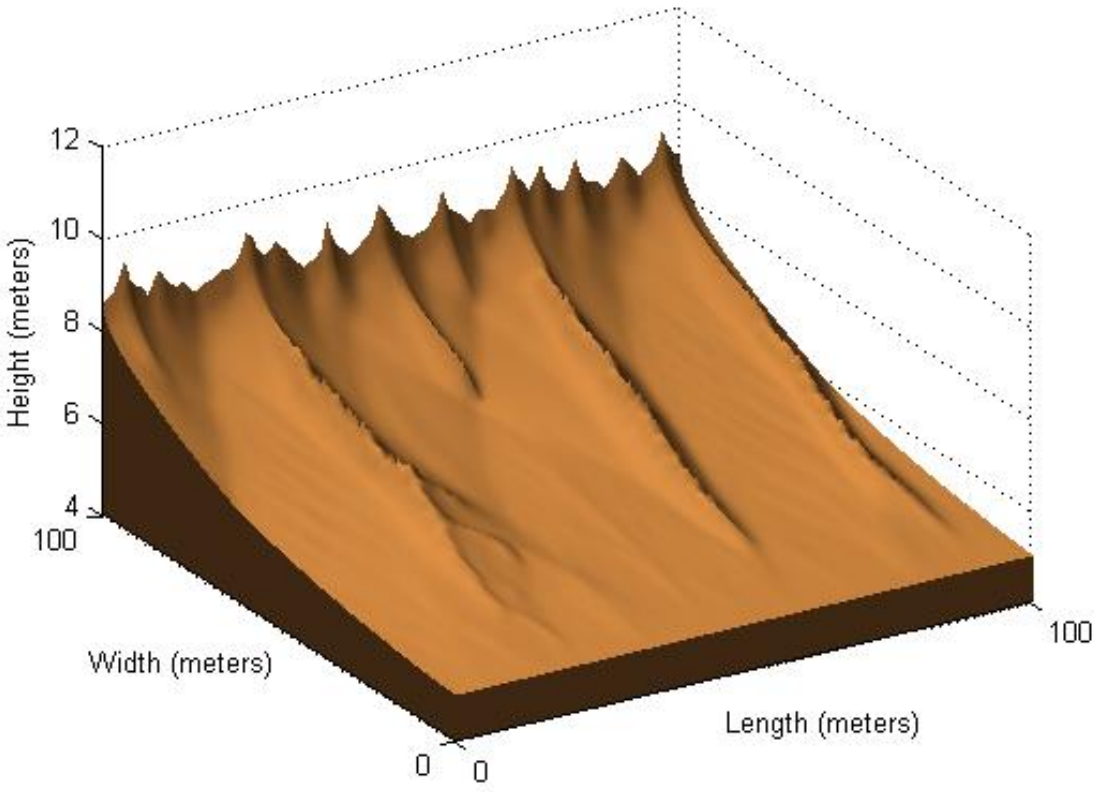


Figure 3.5: A surface with abrasion to dig out channels.

additional term seems to have no adverse effect on the mountain ridge formation, as opposed to the diffusion term. There are three primary mountain ridges separating three river valleys, as expected. Furthermore, these initial abrasive surfaces also generated the expected small scale results discussed in Sect. 2.3.

3.3 Rocks

Another eventual goal is to modify the Smith and Bretherton model to bridge the gap between the transport-limited and detachment-limited cases. The detachment-limited

case is a situation in which there is a surface composed of rock and one must wait for it to weather before the resulting sediment can be transported away [37, 36, 38, 50, 35].

A simple test was done where a rock was added to the surface, limiting the amount of sediment present for erosion. For this test, the free water surface H was given by

$$H(x, y, t) = z(x, y, t) + h(x, y, t) + r(x, y, t), \tag{3.4}$$

where z was still the *erodible* land surface (sediment), h the depth of water over the surface, and r the rock underneath the land surface. The weathering of the rock was simulated using a diffusion equation,

$$\frac{\partial r}{\partial t} = D(x, y)\nabla^2 r, \tag{3.5}$$

where $D(x, y)$ is a diffusion function given by

$$D(x, y) = \begin{cases} D_1 & z(x, y) > 0, \\ D_2 & z(x, y) = 0, \end{cases} \tag{3.6}$$

for positive constants D_1 and D_2. The purpose of the different coefficients was to have the rock weather faster where water was actively flowing against it as opposed to rock that was buried under sediment. The secondary purpose was the digging of rivers. By having this varying diffusion coefficient, the idea was that this would allow the rivers to dig themselves into the rock instead of just digging into sediment.

No changes where made to Eqs. (2.5) or (2.6), but a small change was made to the numerics for the sediment flow equation. After every erosion step, the value of H was

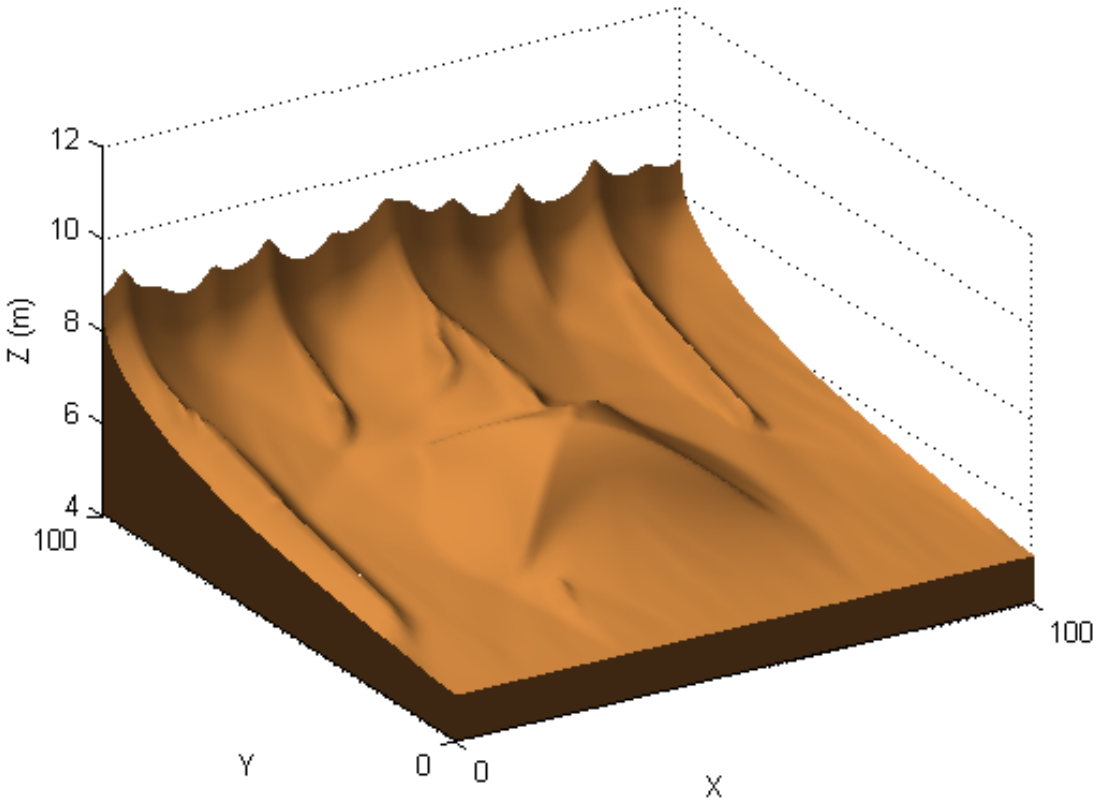


Figure 3.6: A surface with a large boulder under the sediment.

forced to satisfy the natural constraint

$$H(x, y, t) \geq h(x, y, t) + r(x, y, t). \tag{3.7}$$

If the free water surface H fell below this threshold at any point, the value at the point was set to $h + r$. Enforcing this condition in this manner violates the conservation of mass, however this test was a proof of concept as opposed to a realistic simulation.

For the initial test, the rock was given by

$$r(x, y) = 5.0 + 2.0 \cos\left(\frac{\max\{|x - 50.5|, |y - 50.5|\}}{50.5}\pi\right), \tag{3.8}$$

which creates a mass of rock in the center of the surface that tapers below the level were any erosion should occur. Figure 3.6 shows the resulting surface of this test. The diffusion rates were too low for this timescale or erosion, but succeeds as a proof of concept for future work.

3.4 Notable Observation

If an initial surface already contained a river channel, the Smith and Bretherton model can be used to model its evolution [11]. The goal of the abrasion and diffusion terms, however, was to model the formation of river channels in an unchannelized surface by forcing sediment to move in a more realistic manner. Since these tests, it has been observed that modifying the water flow equation is also crucial to the development of river channels.

Consider the conservation of water equation in the case where (i) the water depth is at equilibrium and (ii) where $\mathbf{u}_w$, the direction of flow, is the downstream (y) direction. Then Eq. (2.1) simplifies to

$$R = -\frac{\partial}{\partial y} q_w \tag{3.9}$$

where the flux of water q_w is a function of y. Integrating in the y-direction yields that

$$Ry = C(x) - q_w(y) \tag{3.10}$$

where $C(x)$ is some function of a single variable. In particular, this function $C(x)$ represents the general shape of the channel in the lateral direction. Thus future work must also modify Eq. (2.5) to see more defined river channels in simulations.

Chapter 4

Fractal Interpolation as Small Scale Dynamics

Fractals and methods to construct them, such as iterated functions systems [3], have gained significant popularity in the past few decades. Fractals and fractal functions have developed uses in a wide variety of fields such as number theory, chaos theory, and complex analysis [18], stochastic processes [47], image compression [4, 21], music [65, 31], and terrain modeling [32, 14]. Of great importance to a fractal is its associated fractal dimension [20] and its uses for classifying fractals [51]. There are a number of ways to compute the fractal dimension: using the power spectra or variogram to calculate the dimension directly [68], or computing the Hölder exponent to estimate the dimension [7].

There is a useful relationship between the fractal dimension and the Hurst exponent (see Sect. 4.2). For this reason, it is believed that a relatively new technique of fractal interpolation can be used to force the appropriate small scale dynamics onto a surface. Below is a description of fractal interpolation functions, their construction, and the adaptation of this theory necessary for its use in the current fluvial landscape theory.

4.1 Fractal Interpolation Functions

Fractal interpolation functions [2] are specific cases of *iterated function systems* [3]. An iterated function system (IFS), denoted as $\{X, \omega_n \colon n = 1, 2, \ldots, N\}$, consists of a complete metric space (X, d) and a finite set of continuous mappings $\omega_n : X \to X$ for $n = 1, 2, \ldots, N$. From the initial space X, the space $\mathcal{H}(X) = \{K \subseteq X \mid K \text{ is compact and non-empty}\}$ is considered and equipped with the Hausdorff metric. Define the map $W : \mathcal{H}(X) \to \mathcal{H}(X)$ by

$$W(B) = \bigcup_{n=1}^{N} \omega_n(B). \tag{4.1}$$

If each ω_n is a contraction, then the IFS is called *hyperbolic*, and in this case, the map W has a unique fixed point $A \in \mathcal{H}(X)$ called the *attractor* of the IFS [3].

A fractal interpolation function (FIF) is an iterated function system where the functions ω_n are defined such that the resulting attractor A is the graph of a continuous function that interpolates a given set of data points. A sample construction is given for data in the plane.

Let $\{(x_n, y_n) \in \mathbb{R}^2 \colon n = 0, 1, 2, \ldots, N\}$ be such that $x_0 < x_1 < \ldots < x_N$. Let $I = [x_0, x_N]$, and for $n = 1, 2, \ldots, N$, define $I_n = [x_{n-1}, x_n]$, and let $L_n : I \to I_n$ be contractive homeomorphisms such that

$$\begin{aligned} L_n(x_0) = x_{n-1}, \qquad & L_n(x_N) = x_n, \\ |L_n(u_1) - L_n(u_2)| \leq \ell_n \cdot |u_1 - u_2| \qquad & \forall u_1, u_2 \in I, \end{aligned} \tag{4.2}$$

where $0 \leq \ell_n < 1$.

Let $a < b$ be (finite) real numbers such that $y_n \in [a, b]$ for all $n = 0, 1, 2, \ldots, N$. Define $X = I \times [a, b] \subset \mathbb{R}^2$ and let $F_n : X \to [a, b]$ be continuous such that

$$\begin{aligned} F_n(x_0, y_0) &= y_{n-1}, \qquad F_n(x_N, y_N) = y_n, \\ |F_n(u, v_1) - F_n(u, v_2)| &\leq \xi_n \cdot |v_1 - v_2| \end{aligned} \tag{4.3}$$

for all $u \in I$ and $v_1, v_2 \in [a, b]$, and where $0 \leq \xi_n < 1$. Lastly, define $w_n : X \to X$ by

$$w_n \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} L_n(x) \\ F_n(x, y) \end{bmatrix}. \tag{4.4}$$

Then the set $\{X, w_n \colon n = 1, 2, \ldots, N\}$ is the desired IFS. This iterated function system may not be hyperbolic, but it still has a unique attractor.

Theorem 1 ([2, Thm. 1])**.** *The IFS $\{X, w_n \colon n = 1, 2, \ldots, N\}$ defined above has a unique attractor G. G is the graph of a continuous functions $f : I \to [a, b]$ which obeys*

$$f(x_n) = y_n \qquad \text{for } n = 0, 1, 2, \ldots, N.$$

Proof. See [2]. □

The requirements given in equations (4.2) and (4.3) are extremely broad. The most widely studied FIFs have the general form

$$L_n(x) = a_n x + d_n, \tag{4.5}$$

$$F_n(x, y) = s_n y + q_n(x), \tag{4.6}$$

where a_n and d_n are defined by Eqs. (4.8) and (4.9) given below, $|s_n| < 1$ are free parameters, and $q_n(x)$ are suitably chosen functions so that the $F_n(x, y)$ meet the requirements

of Eq. (4.3). The functions q_n define the general shape of the resulting interpolation function, and thus are often referred to as the *condensation functions* [2]. The affine transformation, Eq. (4.5), allows for the use of the following theorem.

Theorem 2 ([2, Thm. 4])**.** *Let* $\{X, \omega_n \colon n = 1, 2, \cdots, N\}$ *be a hyperbolic IFS, which generates a fractal interpolation function* f *with graph* G *and such that the mappings* $L_n : I \to I_n$ *are affine with* $L_n(x) = a_n x + d_n$*. Let bounds on the contractivities be expressed by numbers* $0 < t_n \leq s_n < 1$ *such that for each* n

$$t_n \cdot d(x, y) \leq d(\omega_n(x), \omega_n(y)) \leq s_n \cdot d(x, y) \quad \textit{for all } x, y \in X,$$

where $d(\cdot, \cdot)$ *is a metric on* X *equivalent to the Euclidean metric. Then the Hausdorff-Besicovitch dimension* D *of* G *is bounded by*

$$\min\{2, \ell\} \leq D \leq u,$$

where ℓ *and* u *are the positive solutions of*

$$\sum_{n=1}^{N} t_n^{\ell} = 1 \qquad \textit{and} \qquad \sum_{n=1}^{N} s_n^{u} = 1;$$

and where for the lower bound to hold it must also be true that

$$t_1 \cdot t_N \leq (\min\{a_1, a_N\}) \left(\sum_{n=1}^{N} t_n^{\ell} \right)^{2/\ell}$$

Proof. See [2] □

As noticed by Barnsley [2], the bounds on the fractal dimension provided by this theorem may be less than ideal. The proposed solution is to iterate the map further, i.e., if

G is the graph of the system $\{X, \omega_n \colon n = 1, 2, \ldots, N\}$ then it is also the graph of $\{X, \omega_n \circ \omega_m \colon n, m = 1, 2, \ldots, N\}$. This iteration technique allows for better bounds on ℓ and u, but is time consuming and may not be precise enough in situations where determining the fractal dimension exactly is of importance.

4.1.1 Affine Fractal Interpolation

Among the most common fractal functions used in practice are *linear* or *affine* fractal interpolation functions, so named because each $\omega_n : X \to X$ is an affine transformation. For the data set $\{(x_n, y_n) \in \mathbb{R}^2 \colon n = 0, 1, 2, \ldots, N\}$, define the ω_n by

$$w_n \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_n & 0 \\ c_n & s_n \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_n \\ e_n \end{bmatrix} \tag{4.7}$$

where

$$a_n = \frac{x_n - x_{n-1}}{x_N - x_0}, \tag{4.8}$$

$$d_n = \frac{x_N x_{n-1} - x_0 x_n}{x_N - x_0}, \tag{4.9}$$

$$c_n = \frac{y_n - y_{n-1}}{x_N - x_0} - s_n \frac{y_N - y_0}{x_N - x_0}, \tag{4.10}$$

$$e_n = \frac{x_N y_{n-1} - x_0 y_n}{x_N - x_0} - s_n \frac{x_N y_0 - x_0 y_N}{x_N - x_0}, \tag{4.11}$$

for $n = 1, 2, \ldots, N$. The s_n are free parameters that must satisfy $|s_n| < 1$ to guarantee that IFS has a unique attractor.

The transformation w_n maps line segments parallel to the y-axis to line segments parallel to the y-axis, contracted by a factor of $|s_n|$. For this reason, the s_n are often referred to as the *vertical scaling factors*, *scaling factors*, or sometimes *contractivity factors*.

Other than its simplicity, another advantage of linear fractal interpolation is the following theorem that allows for a direct calculation of the fractal dimension.

Theorem 3 ([5, Thm. 4])**.** *Let N be a positive integer greater than one. Let $\{(x_n, y_n) \in \mathbb{R}^2 \colon n = 0, 1, 2, \ldots, N\}$ be a set of data. Let $\{X; w_n, n = 1, 2, \ldots, N\}$ be an IFS associated with the data, where*

$$w_n \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_n & 0 \\ c_n & s_n \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_n \\ e_n \end{bmatrix} \qquad \text{for } n = 1, 2, \ldots, N.$$

The vertical scaling factors s_n obey $0 \leq s_n < 1$; and the constants a_n, d_n, c_n, and e_n, are given by Equations (4.8), (4.9), (4.10), (4.11), *for $n = 1, 2, \ldots, N$. Let G denote the attractor of the IFS, so that G is the graph of a fractal interpolation function associated with the data. If*

$$\sum_{n=1}^{N} |s_n| > 1, \tag{4.12}$$

and the interpolation points do not all lie on a single straight line, then the fractal dimension of G is the unique real solution D of

$$\sum_{n=1}^{N} |s_n| a_n^{D-1} = 1. \tag{4.13}$$

Otherwise the fractal dimension of G is one.

Proof. See [5] □

Using only linear fractal interpolation functions may be restrictive, but the benefit of being able to calculate the fractal dimension directly, thanks to Theorem 3, may be of use compared to the estimates of Theorem 2 (see, for example [51]). Unfortunately, the fractal dimension can only be solved for analytically if the a_n are all equal.

Consider (without loss of generality) the case where $0 = a_0 < a_1 < \cdots < a_N = 1$, with the $a_n = 1/N$ for all n. If the scaling factors satisfy $|s_n| = s$ for all n with

$$\frac{1}{N} < s < 1, \tag{4.14}$$

then Barnsley and Harrington [6] noticed that the fractal dimension of the resulting FIF is given by

$$D = 2 + \log_N(s) \tag{4.15}$$

More generally, if the scaling factors are not equal,

$$\sum_{n=1}^{N} |s_n| \left(\frac{1}{N}\right)^{D-1} = \left(\frac{1}{N}\right)^{D-1} \sum_{n=1}^{N} |s_n| = 1,$$

or that

$$\sum_{n=1}^{N} |s_n| = N^{D-1}. \tag{4.16}$$

Then the fractal dimension is given by [1]

$$D = 1 + \log_N \left(\sum_{n=1}^{N} |s_n| \right), \tag{4.17}$$

which simplifies to Eq. (4.15) when the vertical scaling factors s_n are equal in absolute value and satisfy Eq. (4.14).

4.2 Hurst Fractal Interpolation

Solving equation (4.13) algebraically can only be done in the cases shown above: when the abscissa are all equally spaced. The purpose of this section is to introduce a method

of using Theorem 3 in reverse: to find the vertical scaling factors s_n from the desired fractal dimension D of the resulting graph.

One approach is to have all the vertical scaling factors be equal. If $s_n = s$ for all n, then one can solve Eq. (4.13) for s to obtain that

$$s = \left(\sum_{n=1}^{N} a_n^{D-1} \right)^{-1}. \tag{4.18}$$

The fractal dimension $D \in [1, 2)$ and $a_n < 1$, thus $a_n < a_n^{D-1} < 1$. Therefore the value of s is less then one and satisfies Eq. (4.12). Then by Theorem 3, the resulting IFS has a unique attractor with fractal dimension D.

The primary disadvantage of Eq. (4.18) is the initial assumption: forcing all the vertical scaling factors to be the same. One might expect that for a self-affine fractal, the horizontal and vertical scalings of a section should be more closely related to one another. This is one of the benefits of the following corollary: it produces a fractal function of a desired fractal dimension by relating the vertical scaling of a section with its horizontal scaling.

Corollary 1. *Let* $\{(x_n, y_n) \in \mathbb{R}^2 : n = 0, 1, 2, \ldots, N\}$ *be a set of noncollinear data where* N *is a positive integer greater than one. Let* $\{X; w_n, n = 1, 2, \ldots, N\}$ *be an IFS associated with the data, where*

$$w_n \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_n & 0 \\ c_n & s_n \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_n \\ e_n \end{bmatrix} \qquad \text{for } n = 1, 2, \ldots, N,$$

and where the constants $a_n, d_n, c_n,$ *and* $e_n,$ *are given by Equations* (4.8), (4.9), (4.10),

(4.11)*, respectively. If* $1 < D < 2$ *and*

$$|s_n| = a_n^{2-D} \tag{4.19}$$

for each $n = 1, 2, \ldots, N$*, then the resulting attractor* G *has fractal dimension* D*.*

Proof. From Equation (4.8) it is clear that $0 < a_n < 1$ for all n and that

$$\sum_{n=1}^{N} a_n = 1. \tag{4.20}$$

The quantity $2-D$ is positive and less than one. Combined with the above observation(s) yields that

$$1 = \sum_{n=1}^{N} a_n < \sum_{n=1}^{N} a_n^{2-D} = \sum_{n=1}^{N} |s_n|.$$

Thus this IFS satisfies the criteria of Theorem 3.

Let D_G be the fractal dimension of the G, the attractor of the IFS. Then D_G is the unique real solution of

$$\sum_{n=1}^{N} |s_n| a_n^{D_G-1} = 1.$$

By the choice of the vertical scaling factors,

$$1 = \sum_{n=1}^{N} |s_n| a_n^{D_G-1} = \sum_{n=1}^{N} a_n^{1+D_G-D}.$$

By Eq. (4.20),

$$\sum_{n=1}^{N} a_n = \sum_{n=1}^{N} a_n^{1+D_G-D}.$$

These finite sums can be re-arranged to get that

$$\sum_{n=1}^{N} a_n \left(1 - a_n^{D_G - D}\right) = 0. \tag{4.21}$$

If $D_G > D$, then the summation in Eq. (4.21) contains strictly positive terms, and thus can not be equal to zero. Similarly if $D_G < D$, then $a_n^{D_G - D} > 1$, and the sum contains strictly negative terms. Therefore, it must be the case that $D_G = D$. □

This corollary is a generalization of the results discussed at the end of Section 4.1.1. If the $a_n = 1/N$ and $|s_n| = s$ for all n, then Eq. (4.16) reduces to

$$N^{D-1} = \sum_{n=1}^{N} |s_n| = N \cdot s$$

which implies that

$$s = N^{D-2} = \left(\frac{1}{N}\right)^{2-D} = a_n^{2-D}.$$

For a fractal with topological dimension n, the quantity $H_u = n + 1 - D$ is the Hurst exponent [43, 65], hence the name given to this particular method of specifying the desired fractal dimension (or Hurst exponent) to create a FIF. The Hurst exponent is the defining characteristic of fractional Brownian motions [42, 43]. Thus Corollary 1 presents a method for creating a H_u-fractional Brownian motion (fBm) that interpolates any set of initial data; regardless of the spacing of its abscissa. It is worth noting that Malysz [47] used fractal interpolation to approximate stochastic processes and fBms using data at uniform spacing and found convergence almost surely to the original process.

Furthermore, this is the same Hurst exponent discussed previously in Sect. 2.3. The goal is thus to use Hurst fractal interpolation to force a surface to have the appropriate scaling results. See Sects. 4.3 and 5.2.5 for more specifics on this process.

4.2.1 Vector-Valued Interpolants

There exists an analogous result to Theorem 3 for a specific class of affine fractal interpolation functions in higher dimensions. Below are the function requirements and the main theorem needed for the Hurst (vector-valued) fractal interpolation functions. For more of the background on vector-valued fractal interpolation functions, see [45].

Without loss of generality, let $\{(t_n, \mathbf{x}_n) \in I \times \mathbb{R}^m : n = 0, 1, \ldots, N\}$ be the given data set where $I = [0, 1] \subset \mathbb{R}$ is the unit interval and $0 = t_0 < t_1 < \cdots < t_N = 1$ is a partition of the interval. Define

$$h_n(t) = a_n t + t_{n-1} \tag{4.22}$$

where $a_n = t_n - t_{n-1}$, and define

$$\Psi_n(t, \mathbf{x}) = A_n \begin{bmatrix} t \\ \mathbf{x} \end{bmatrix} + \mathbf{d}_n, \tag{4.23}$$

where $A_n : \mathbb{R}^{m+1} \to \mathbb{R}^m$ is given by

$$A_n = \begin{bmatrix} c_{n,1} & s_{n,1} & 0 & \cdots & 0 \\ c_{n,2} & 0 & s_{n,2} & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ c_{n,m} & 0 & \cdots & 0 & s_{n,m} \end{bmatrix},$$

with

$$c_{n,k} = [\mathbf{x}_n]_k - [\mathbf{x}_{n-1}]_k - s_{n,k}\Big([\mathbf{x}_N]_k - [\mathbf{x}_0]_k\Big),$$

and

$$[\mathbf{d}_n]_k = [\mathbf{x}_{n-1}]_k - s_{n,k}[\mathbf{x}_0]_k,$$

where $[\mathbf{x}_n]_k$ is the k^{th} component of $\mathbf{x}_n$. The $s_{n,k}$ are free parameters that must satisfy $0 \leq |s_{n,k}| < 1$ for all $n = 1, 2, \ldots, N$ and $k = 1, 2, \ldots, m$.

Theorem 4 ([45, Thm. 1])**.** *Let $\mathbf{W}_n(t, \mathbf{x}) = (h_n(t), \Psi_n(t, \mathbf{x}))$ where h_n and Ψ_n are defined as in* (4.22) *and* (4.23)*, respectively. Let G be the graph of the vector-valued fractal interpolation function generated by the IFS. Let $\lambda_n = \prod_{k=1}^{m} |s_{n,k}|$, and suppose that the set of interpolation points is not contained in any hyperplane of $\mathbb{R}^{m+1}$.*

(a) If $\sum_{n=1}^{N} \lambda_n > 1$, then the fractal dimension of G is D, where D is the unique positive solution of

$$\sum_{n=1}^{N} \lambda_n a_n^{D-m} = 1. \tag{4.24}$$

(b) If $\sum_{n=1}^{N} |s_{n,k}| \leq 1$ for all k, then the fractal dimension of G is 1.

(c) Suppose that $\sum_{n=1}^{N} \lambda_n \leq 1$ and that there exists an h such that $\sum_{n=1}^{N} |s_{n,k}| > 1$, for $k \in \{1, \ldots, h\}$ and $\sum_{n=1}^{N} |s_{n,k}| \leq 1$, for $k \in \{h+1, \ldots, m\}$. Denote by $\mathbf{p}$ a p-tupel of elements from $\{1, \ldots, h\}$, and let $\lambda_n^{(p)} = \prod_{k \in \mathbf{p}} |s_{n,k}|$. Let $\mathbf{q} = \max\{\mathbf{p} \colon \mathbf{p} \subseteq \{1, \ldots, h\}\}$ such that $\sum_{n=1}^{N} \lambda_n^{(p)} > 1$. Then the fractal dimension of G is

$$D = \max\{d^{(q)} \colon \mathbf{q} \subseteq \{1, \ldots, h\}\},$$

where $d^{(q)}$ is the unique positive solution of

$$\sum_{n=1}^{N} \lambda_n^{(q)} a_n^{d^{(q)}-m} = 1.$$

If $\{(t_n, \mathbf{x}_n)\} \subseteq H^{\mu}$, where H^{μ} is a hyperplane of codimension μ in $\mathbb{R}^m$, $1 \leq \mu \leq m-1$, then conclusions (a)-(c) hold with m replaced by $m - \mu$.

Finally, if $\{(t_n, \mathbf{x}_n)\} \subseteq H^m$, then the fractal dimension of G is 1.

Proof. See [45] □

Corollary 2. *Let* $\mathbf{W}_n(t, \mathbf{x}) = (h_n(t), \Psi_n(t, \mathbf{x}))$ *where* h_n *and* Ψ_n *are defined as in* (4.22) *and* (4.23)*, respectively. If* $m \leq D < m + 1$ *and*

$$\prod_{k=1}^{m} |s_{n,k}| = a_n^{1+m-D} \tag{4.25}$$

for all $n = 1, 2, \ldots, N$*, then the fractal dimension of the attractor* G *is* D*.*

Proof. If (4.25) holds for all n then

$$1 = \sum_{n=1}^{N} a_n < \sum_{n=1}^{N} a_n^{1+m-D} = \sum_{n=1}^{N} \left(\prod_{k=1}^{m} |s_{n,k}| \right) = \sum_{n=1}^{N} \lambda_n,$$

and thus this IFS falls into case (a) of Theorem 4. The same argument as in Corollary 1 completes the proof. □

4.2.2 Hidden Variable Interpolants

Lastly, the idea of Hurst interpolation is extended to hidden variable fractal interpolation functions, which were introduced by Barnsley et al. [5]. The idea behind a hidden variable FIF is to construct an interpolant in a higher dimensional space and then project the graph onto a lower dimensional space. The creation of the interpolant in a higher dimensional space allows for more degrees of freedom that become "hidden" when the graph is projected. As with the previous section, enough detail is given below to formulate the main theorem and its corollary. For more background on hidden variable fractal interpolation, see [5, 1, 46].

Let I be defined as above and (Y, d_Y) be a complete metric space. Assume further that Y is a finite direct sum of linear subspaces. Let $\{(t_n, y_n)\} \subset I \times Y$ be a given set of data with $0 = t_0 < t_1 < \cdots < t_N = 1$. Let $h_n(t)$ be defined as in Eq. (4.22) and let $T_n : Y \to Y$ satisfy

$$T_n(y_0) = y_{n-1} \qquad \text{and} \qquad T_n(y_N) = y_n,$$

with the additional constraint that

$$d_Y(T_n(y), T_n(y')) = s_n \cdot d_Y(y, y') \tag{4.26}$$

for all $y, y' \in Y$, with $0 \leq s_n < 1$. Then the IFS is given by $\{I \times Y, W_n \colon n = 1, 2, \ldots, N\}$ where

$$W_n(t, y) = \begin{bmatrix} h_n(t) \\ T_n(y) \end{bmatrix}. \tag{4.27}$$

It is of importance to note that this system is hyperbolic with the proper metric on the product space $\mathbb{R} \times Y$. That metric is defined by

$$d\Big((t, y), (t', y')\Big) = |t - t'| + \theta d_Y(y, y') \tag{4.28}$$

where $\theta > 0$ is determined by the IFS [5, 1, 46].

Let $F : I \to Y$ be the generated fractal interpolation function and let $f_k : I \to Y_k$ be the projection of F onto Y_k, one of the linear subspaces in the summand of Y.

Theorem 5 ([46, Thm. 109])**.** *Let $f_k : I \to Y_k$ be a hidden variable fractal function defined above. Suppose that* $\dim Y_k = m$ *and that* $\sum_{n=1}^{N} s_n^m > 1$*. Then the fractal dimension of the graph of f_k is D, the unique positive solution of*

$$\sum_{n=1}^{N} s_n^m a_n^{D-m} = 1. \tag{4.29}$$

Proof. See [5, 26] □

Corollary 3. *Let $f_k : T \to Y_k$ be a hidden variable fractal function defined above. Suppose that* $\dim Y_k = m$*, $m \leq D < m + 1$, and that*

$$s_n = a_n^{1+\frac{1-D}{m}} \tag{4.30}$$

for all $n = 1, 2, \ldots, N$. Then the fractal dimension of the graph of f_k is D.

Proof. The proof is identical to that of Corollary 1. □

These two results are included (i) for the completeness of the theory, and (ii) for their potential use in the theory of fluvial landscapes. The one-dimensional theory of Corollary 1 is used below to (re-)create fluvial surfaces. This approach does cause a few issues with the resulting surface (see Sect. 5.3.2) that could possibly be avoided with either vector-value or hidden variable fractal interpolation functions. The next section (and chapter) discusses how Corollary 1 can be used to add the necessary small scale features to a landscape.

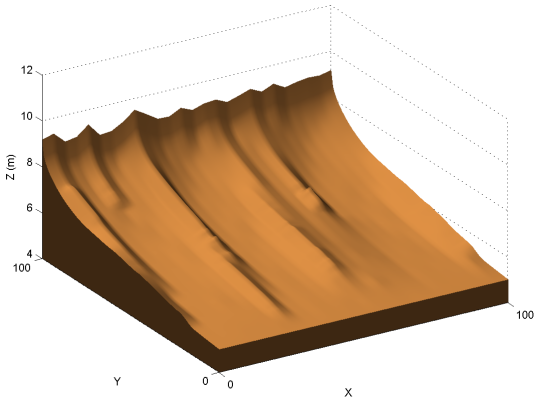
4.3 Application to Fluvial Landscapes

In the case of the Smith and Bretherton model, the variogram is taken in a single dimension, parallel to the base of the ridge. Using Corollary 1, the goal is to add the appropriate small scale dynamics to a surface that already contains the large scales. A value of 0.75 is used as the Hurst exponent as that is the scaling generated by this model.

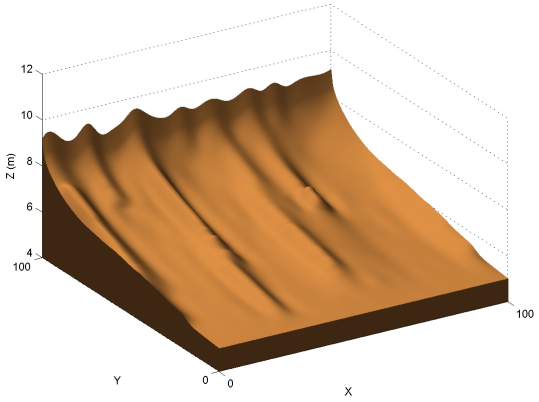
To test this idea, a number of surfaces previously generated by this model, such as the one seen in Fig. 3.1, were chosen. Similar to the work of Chen et al. [14], a few points are selected from each lateral cross-section to represent the general shape of the data. Three different methods were used to select these points: a simple selection of evenly spaced points, the Douglas-Peucker (DP) method of data reduction [16], and an "extrema" selection algorithm described below. Hurst fractal interpolation is then used to restore the removed points. For comparison, (periodic) cubic spline interpolation and linear interpolation are also used to recover the missing data. The use of periodic cubic spline is justified as the lateral boundaries during simulations are periodic (Sect. 2.1.3). The variogram scaling as well as the average percent error between the original surface(s) and their interpolated counter-parts are calculated for comparison purposes.

The first test simply kept every fourth point of the original data. Figure 4.1 shows the reconstructions of Fig. 3.1 using the three interpolation methods. The error and scaling results are given in Fig. 4.2.

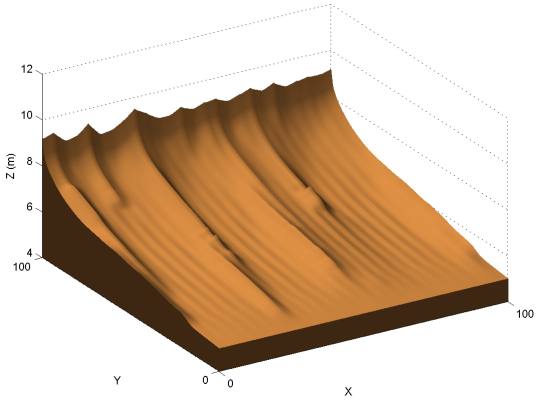
All three methods kept the large scale features of the surface: the three primary mountain ridges and the three valleys separating them. The percentage error in the reconstructed surfaces is relatively low, with the Hurst interpolation technique introducing the most error, which is not surprising considering Fig. 4.1c. The variogram, however, is a more important quantity for the accuracy of these generated surfaces [12].



(a) Linear Interpolation

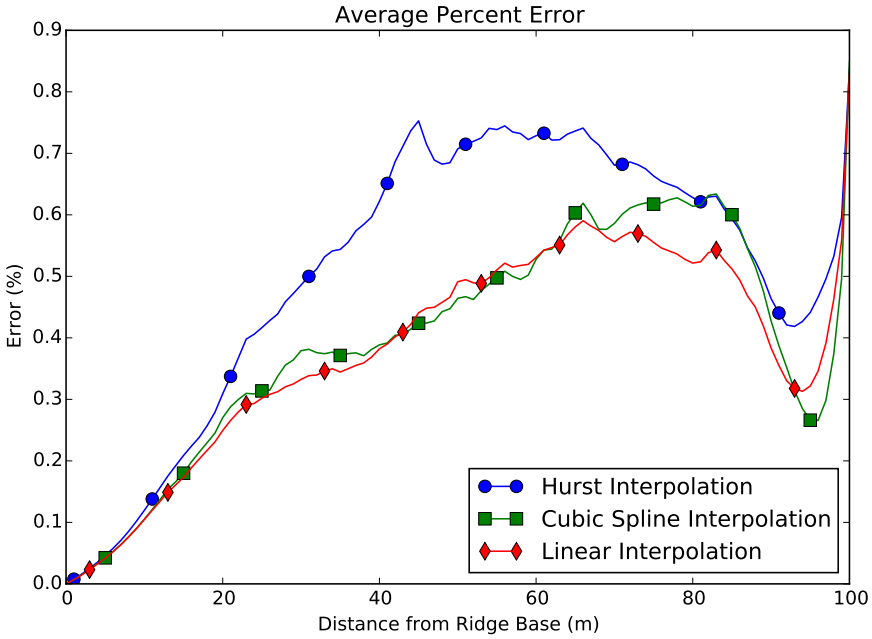


(b) Cubic Interpolation

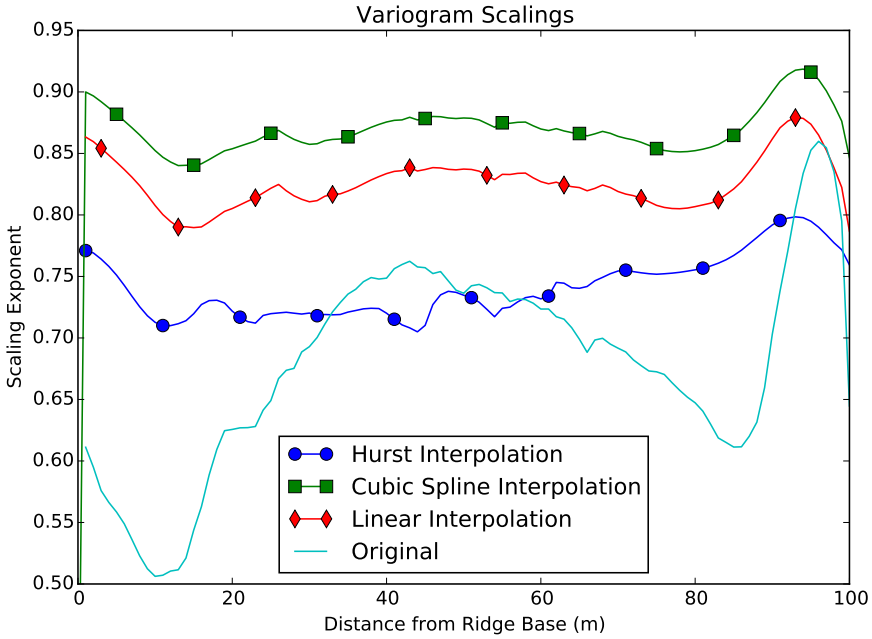


(c) Fractal Interpolation

Figure 4.1: Reconstructed surfaces when retaining every fourth piece of information



(a) Error between interpolations and the originals



(b) Variogram Scalings

Figure 4.2: Results when retaining every fourth piece of (lateral) information

As previously mentioned, the boundary conditions at the top and bottom of the model skew the scaling results, making the center the most accurate place to determine the scalings [12]. All three interpolation techniques leveled the scaling results over the entire surface; in a sense, correcting the bias induced by the boundary conditions. The scaling for the Hurst interpolation is exactly as intended, based on Corollary 1. The scalings results for the linear and cubic spline interpolated surfaces, while not at the desired 0.75, are within a reasonable range based on other work, such as that of Voss [65] and Simpson & Schlunegger [57], to name a few.

While using equally spaced points does not demonstrate the full "power" of Corollary 1, it is an obvious starting point as well as a proof of concept. This result does lay the groundwork for new possibilities for solving the Smith and Bretherton model. Namely, this result shows it is feasible to run the model on a coarse grid (with equally spaced grid points), and then use Hurst interpolation to fill in the details and continue the simulation on a finer grid.

Considering the results in Fig. 4.2, a better option for the reconstruction of data might be a cubic fractal interpolation function [13]. These functions have cubic condensation functions $q_n(x)$ designed such that these interpolants converge to the typical cubic spline as the vertical scaling coefficients are sent to zero. These functions would provide the general shape of a cubic interpolant with the needed randomness of the small scales. Unfortunately, Theorem 2 is the only current theory for finding the fractal dimension analytically, or at least an estimate that can be found before the interpolation is executed.

The next two tests are to reconstruct a surface from strategically picked data. In the lateral direction, the retained data was algorithmically selected in an attempt to keep the "feature" points of the surface, the mountain ridges and valleys, using as few points as possible. Two algorithms were used to select these feature points.

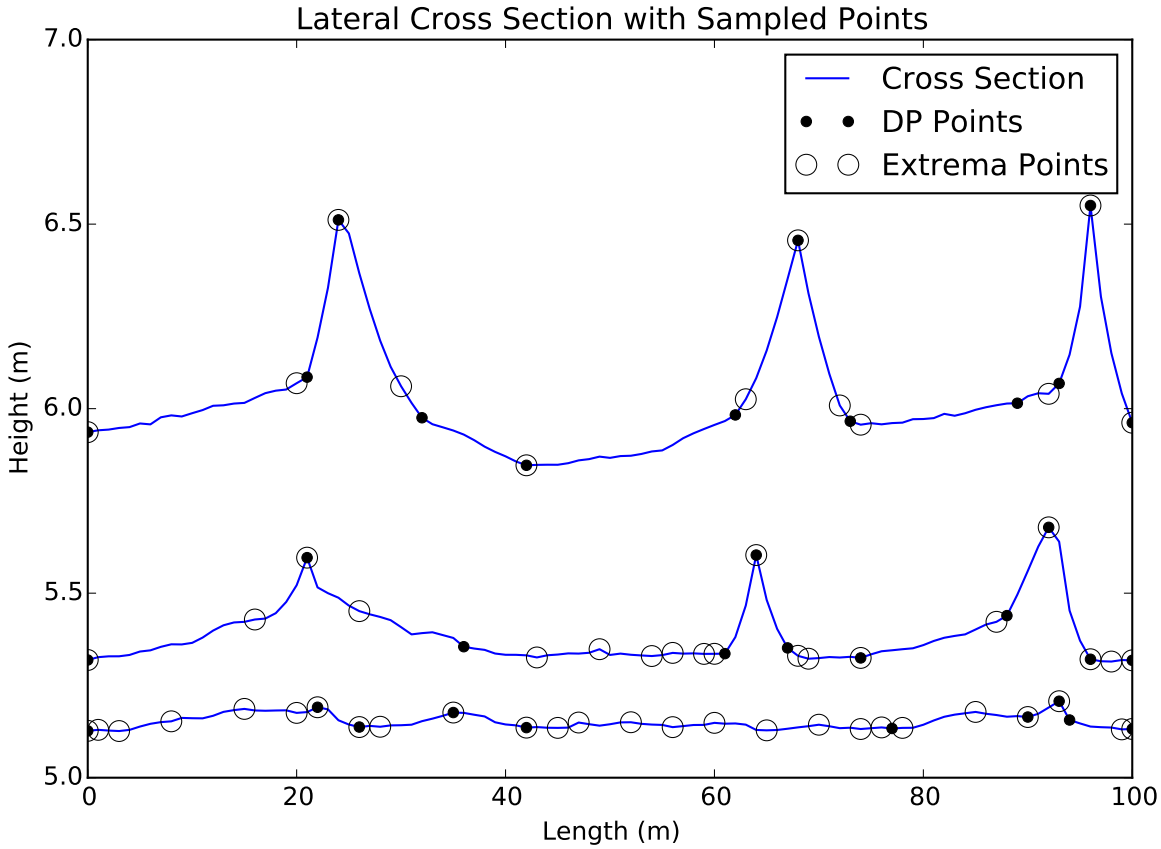


Figure 4.3: Various cross sections of a surface and the selected feature points

The first method is referred to, in this text, as the extrema method. This algorithm begins by using numerical differentiation to find the local extrema, which represent the mountain peaks and valley bottoms. Next is an attempt to find the mountain/valley bases. Starting from each extremum, the algorithm searches for a point to the left and right that best approximates the mountain/valley in the least squares sense. A bias is added to the R-squared value to encourage the selecting of the furthest point possible that best captures the profile's general shape.

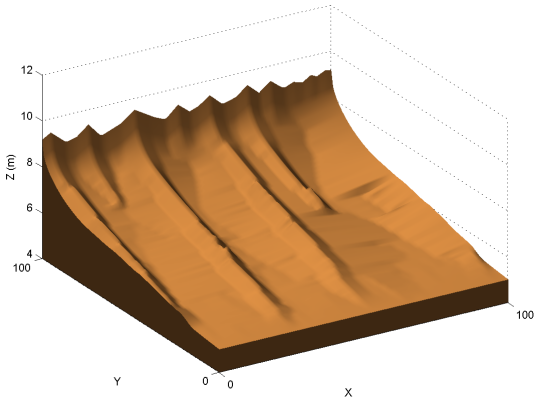
The other method is the (improved) DP algorithm. Using the endpoints to create an initial linear interpolation, the algorithm finds the point with the largest (straight-line)

distance between it and the interpolant. If the distance between that point and the interpolant is greater than some given tolerance ε, then it is selected as an interpolation point. This process repeats until all of the remaining points are within ε of the resulting linear interpolation. For each cross section, the tolerance is set to be a quarter of the distance between the maximum value and the minimum value in that cross section.

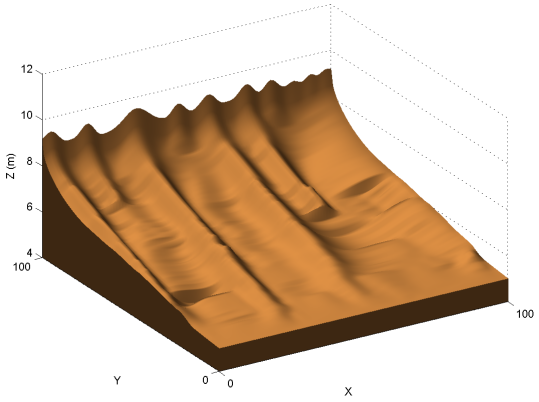
Both algorithms work well at capturing the general shapes of each cross section. The DP method selected an average of 11 points per cross section, while the extrema method selected 19 on average. As can be see in Fig. 4.3, the discrepancy primarily occurs at cross sections closer to the base of the ridge. Near the base of the ridge, the cross sections are extremely jagged, just at a significantly smaller scale, and thus have significantly more local extrema than further up the ridge where the mountains have formed. However, because the variation in heights is small, the DP method does not select as many points.

Figure 4.4 shows the reconstructed surfaces using the extrema method. The error and scaling results for these surfaces are given in Fig. 4.5. Considering how the interpolation points were selected, it is of little surprise how well the linear interpolation method does at reconstructing the surface.

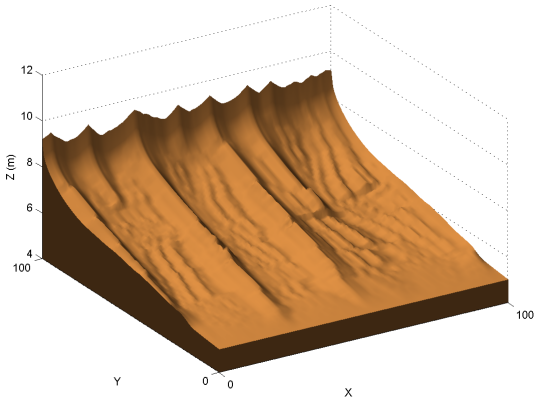
The scalings of the variogram, Fig. 4.5b, are of particular interest. Once again the cubic spline scalings are higher than expected, but still in a reasonable range for natural landscapes. The linear interpolation is surprisingly close to the desired scaling exponent. The error in the Hurst interpolation scaling is explained (initially) by over approximation. In the even space tests, data at the missing locations was calculated exactly by already known points in early iterations. Thanks to the uneven spacing, finding the data at the desired points requires (i) a considerable number of iterations when doing the fractal interpolation, or (ii) approximation from a few iterations. If more iterations of fractal interpolation were performed per cross-section, the resulting scaling exponent



(a) Linear Interpolation

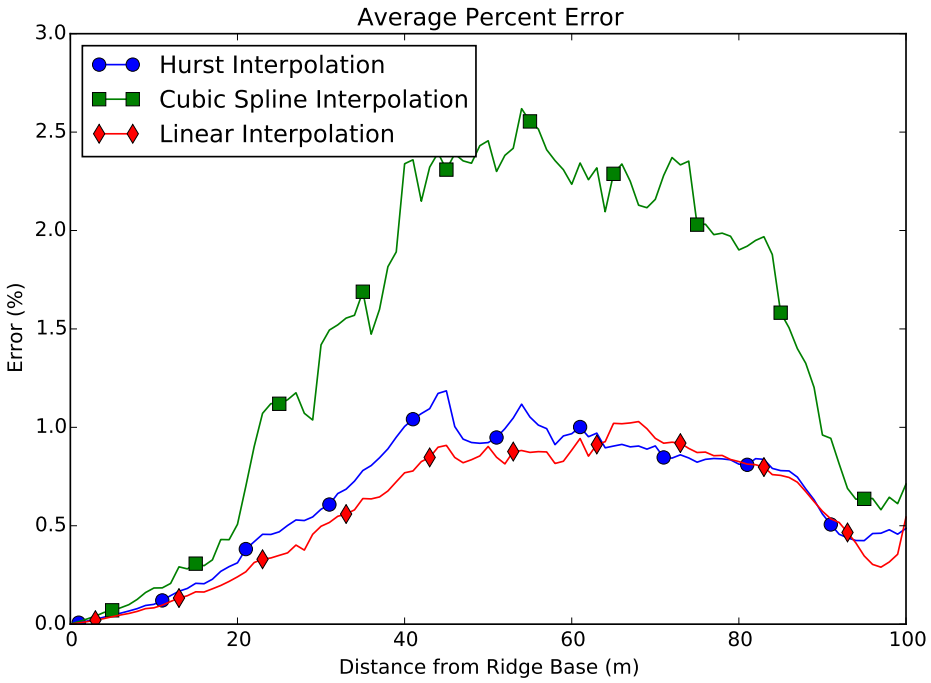


(b) Cubic Interpolation

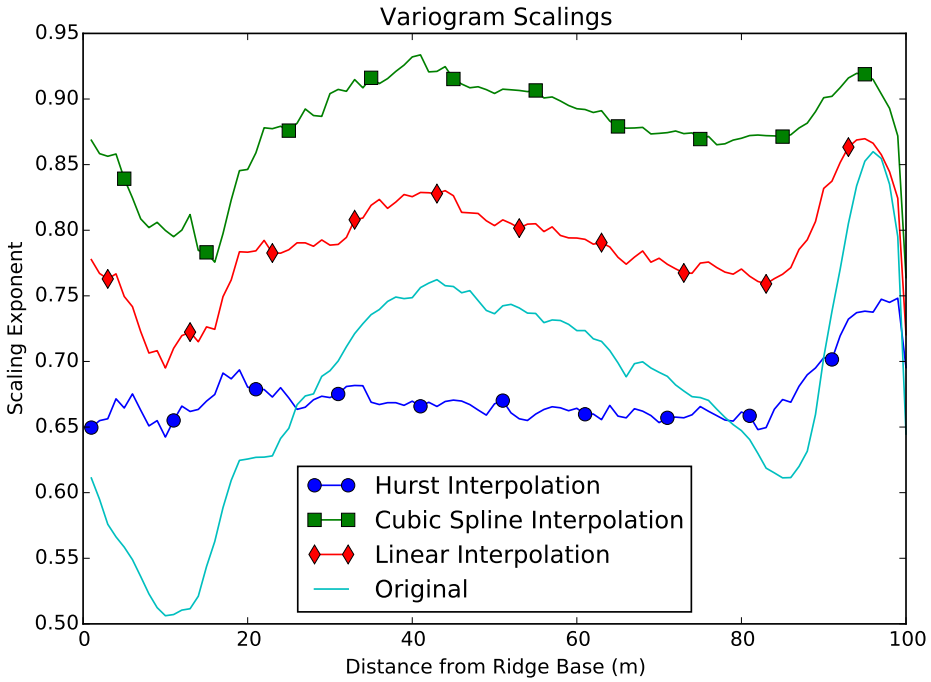


(c) Fractal Interpolation

Figure 4.4: Reconstructed surfaces using the extrema method to select points



(a) Error between interpolations and the originals



(b) Variogram Scalings

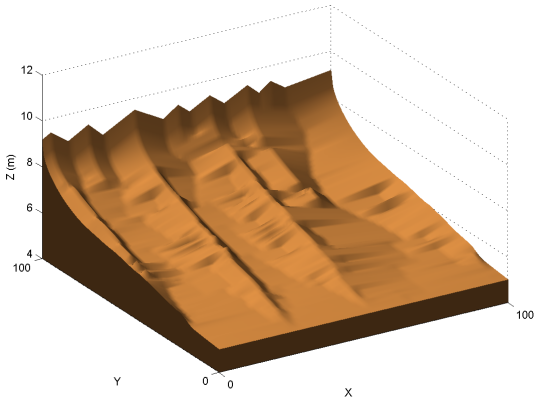
Figure 4.5: Results using the extrema method to select interpolation points

would increase to the expected value (at the trade off of computation time).

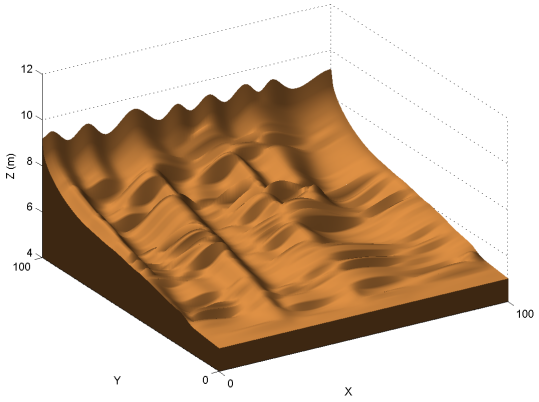
Figure 4.6 shows the the reconstructed surfaces with the DP selection method, and the results can be found in Fig. 4.7. Unsurprisingly, as with the extrema method, the linear interpolation does well at reconstructing the surface, the cubic spline suffers from a lack of data, and the Hurst interpolation has an unexpected variogram scaling.

The Hurst interpolation method replaces the missing data within two percent error of the original surface and significantly steadies the scaling. While the value of the scaling is not exact, it is believed that running these surfaces through the numerics discussed in Sect. 2.2 will both smooth the surface, making them appear more realistic, and amend the scaling error.

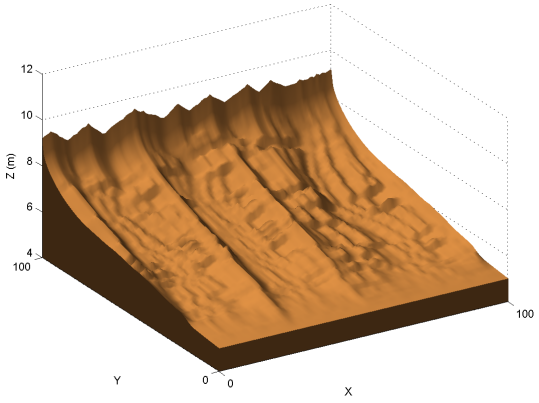
The last thing of interest in the Hurst interpolated surfaces, Figs. 4.4c and 4.6c, are the extra features created within the valleys as a result of the interpolation. These structures vaguely resemble river channels, and it is the hope that these features act as such in numerical simulations.



(a) Linear Interpolation

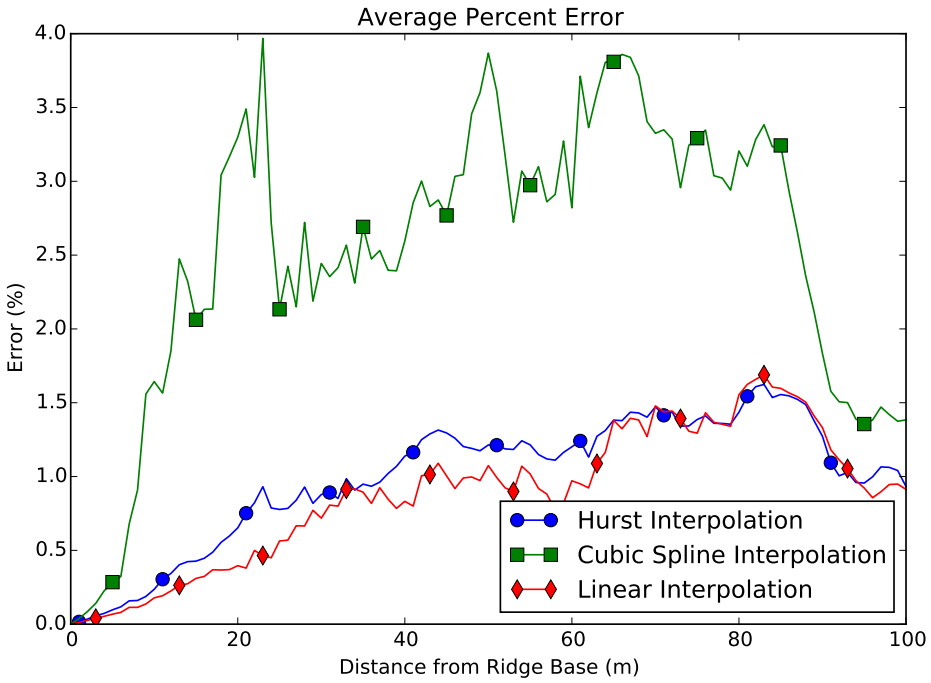


(b) Cubic Interpolation

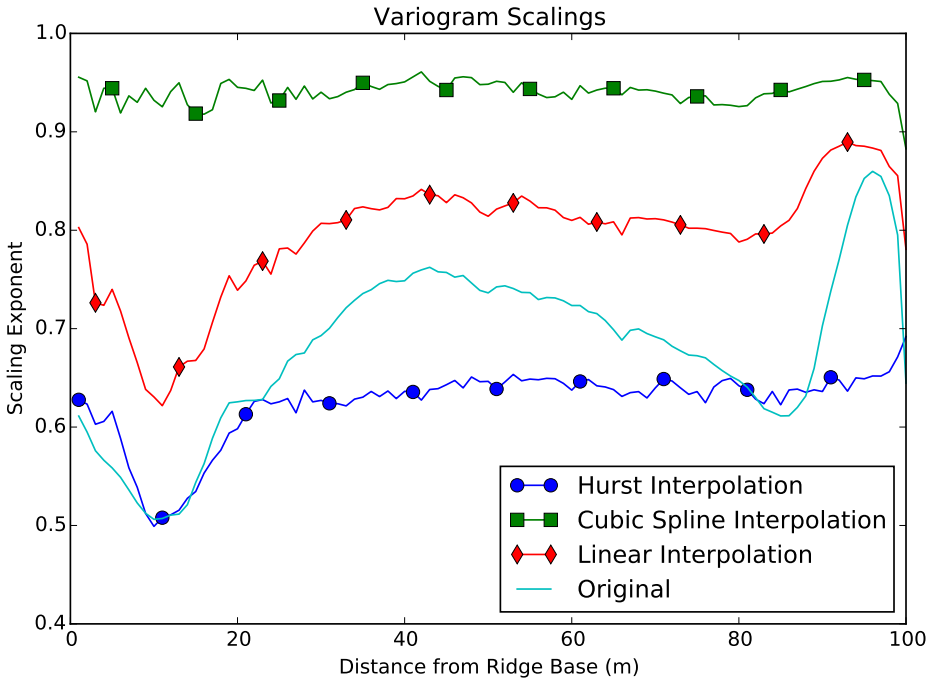


(c) Fractal Interpolation

Figure 4.6: Reconstructed surfaces using the DP method to select interpolation points



(a) Error between interpolations and the originals



(b) Variogram Scalings

Figure 4.7: Results using the DP method to select interpolation points

Chapter 5

Quick Generation Methods

The upwind/Crank-Nicolson code described in Sects. 2.2.2 and 2.2.3 has been implemented on a parallel supercomputer at the University of California, Santa Barbara. The factoring scheme used on Eq. (2.6) allows for easy parallelism, while the upwind scheme is parallelized using a grid tiling method. The mature phase of erosion occurs about when 60% of the initial sediment has been eroded away, and it is during this phase that many of the desired scaling relations discussed in Sect. 2.1.3 become present. Depending on the number of parallel processes used (typically nine per surface, but the code is adaptable), this amount of erosion takes around three months of computation. The goal of this chapter is to consider methods to reduce this computation time.

5.1 Numeric Alterations

Initial ideas on how to speed up numeric computations include: (i) faster numerical method(s), (ii) more processors in parallel computations, (iii) the use of a higher water depth tolerance, or (iv) using a coarser grid. Ideas (i) and (ii) are closely related, as

it is possible that there are more efficient parallel numerical algorithms to solve the model, or even that there is a more efficient way of implementing the current numerics on a parallel infrastructure. Furthermore, the efficiency of most parallel implementations depends on the computer architecture, which can change vastly between machines. Using more processors also has the trade-off of time saved from parallelism versus time spent message passing. For these reasons, focus is placed on modifying the model and its parameters over modifying the algorithms.

For consistency, all tests involving modification of parameters were run on the same supercomputing cluster as the initial tests using nine processors per surface.

5.1.1 Water Equilibrium Tolerance

Solving the water flow equation is the most time consuming piece of the code. The upwind scheme is recursively solved until the water depth reaches an equilibrium. Numerically, this equilibrium is defined as the water depth at every point changing by less than some predefined tolerance between successive iterations. To find the desired tolerance for these simulations, tests were done on Eq. (2.5), comparing the resulting equilibrium water depth for the initial perturbed ridges, see Fig. 5.1. From this data, it was decided that a tolerance of 10^{-8} meters would be used for the model. Based on this data, a tolerance of 10^{-7} would be sufficient for numerical simulations, however no significant decrease in computation time is expected by raising the water depth tolerance to 10^{-7} from 10^{-8} based on the data.

Every time Eq. (2.6) is solved numerically, the water depth is equilibrated again. Considering that such a small amount of sediment can be moved from any one location, one would expect the water depth to not change by a significant amount. It is not the amount

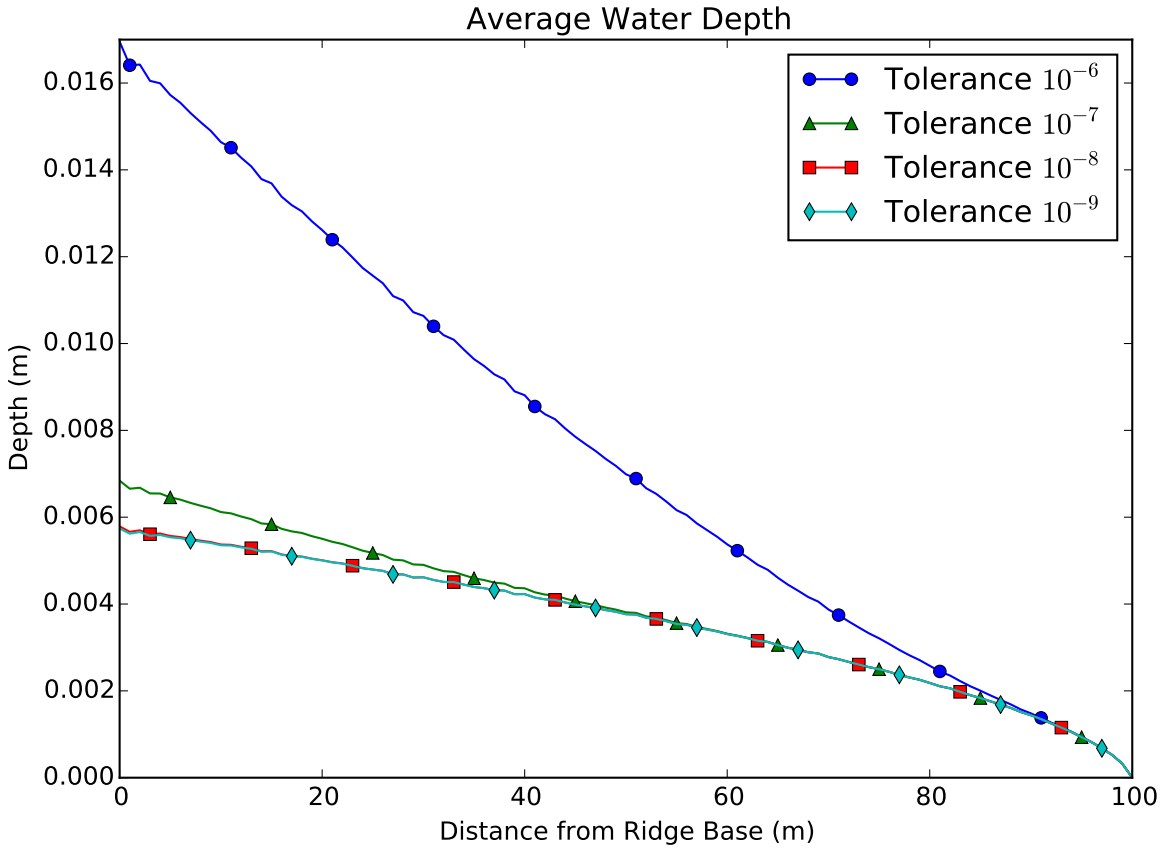
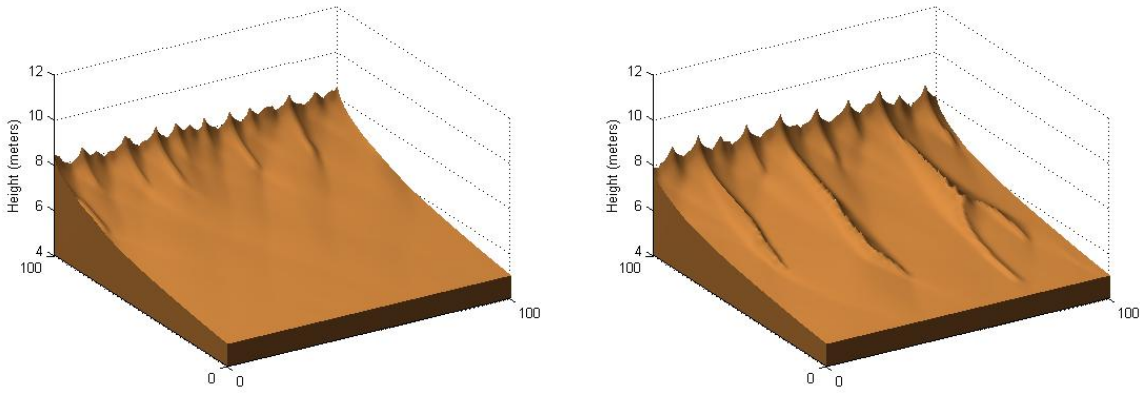


Figure 5.1: Equilibrium water depths per lateral cross section for different tolerances

the depth changes by, but the number of iterations it requires to equilibrate. When the tolerance is 10^{-8}, an average of 20,000 iterations of Eq. (2.5) are performed until the new equilibrium is achieved. With a tolerance of 10^{-6}, the average drops to a mere 1,000 iterations.

Due to this drastic decrease in iterations, the code was run with 10^{-6} as the tolerance. From an unchannelized linear ridge to the target of 60% eroded took a matter of hours with such a high tolerance, instead of months. As illustrated by Fig. 5.2, however, there are some consistency issues with such a high tolerance.



(a) The expected ridges prematurely eroded (b) The ridges remain as expected

Figure 5.2: Two surface generated with a tolerance of 10^{-6}.

It is known that the random perturbations on the initial surface are needed to drive the erosion process [62, 12, 10]. These perturbations play an even more crucial role in the erosion process with such a high tolerance. Both initial surfaces in Fig. 5.2 started with initial perturbations of up to a centimeter in height over about 20% of the surface. The only difference is the distribution of these perturbations over the surface.

Larger powers of 10 were experimented with for the water depth tolerance. All of those resulting surfaces turned out as Fig. 5.2a: overly smoothed.

5.1.2 Non-constant Tolerance

Using a higher tolerance saves a significant amount of computation time, but it is not worth having code run for hours to get an undesirable (and unrealistic) result such as Fig. 5.2a. One possible compromise is to alternate between a higher and lower tolerance. The idea goes back to expectation that the water level should not change much be successive erosion steps. Thus it should be sufficient to use a higher tolerance majority of the time, and switch to a lower tolerance after specific milestones of erosion.

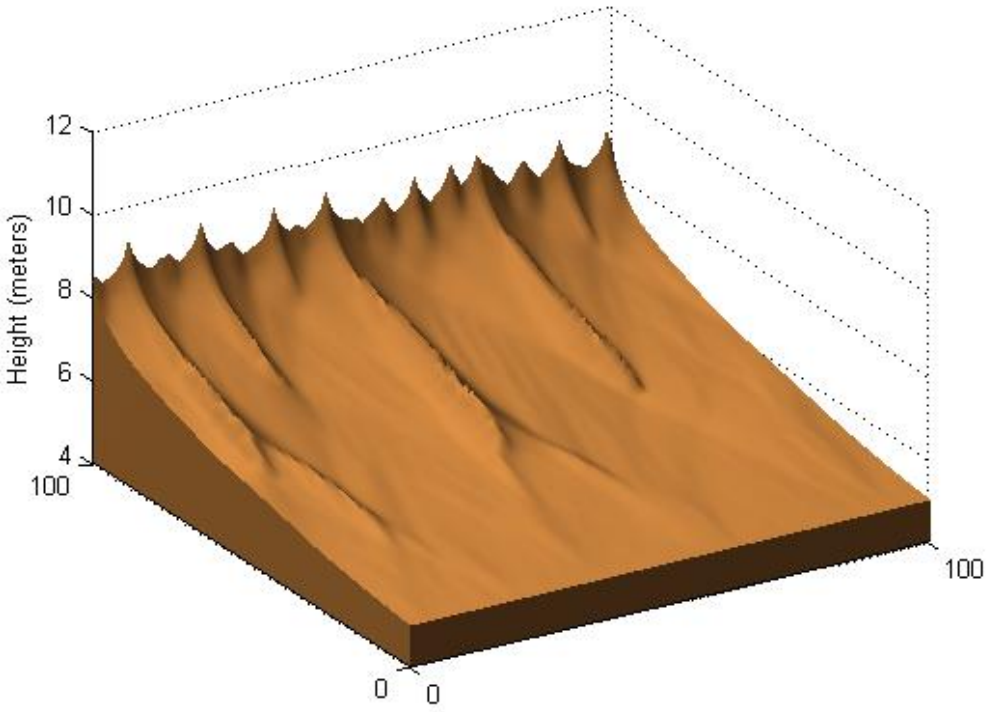


Figure 5.3: A surface where the tolerance alternated between 10^{-6} and 10^{-8}

Figure 5.3 shows a surface generated with an alternating tolerance. At every quarter of a percent eroded, the next water equilibration step was performed with a tolerance of 10^{-8}. Otherwise, the higher tolerance of 10^{-6} was used. Switching tolerances in this manner took roughly a week to reach the 60% eroded milestone, plus or minus a couple days for some surfaces.

Unlike a constant high tolerance, none of the surfaces (thus far) have become overly smooth. The switching to a lower tolerance seems to "flush" the surface of its overabundance of water, preventing the large scale features from eroding away. The success of this method brings into question more localized tolerance changes, such as the possibility of altering the tolerance of particular grid cells based on how much the topography changed in the erosion step.

5.1.3 Coarser Grids

Typically, Eqs. (2.5) and (2.6) are solved over a domain of 100 meters by 100 meters tiled using square grid cells of 1 m by 1 m. This discretization has been under some scrutiny for already being too coarse a grid (see [23], for example). The idea here is to use a coarse(r) grid to get the large scale features and then use Hurst fractal interpolation to fill in the small scale details, creating a multigrid type method.

The previous chapter showed that Hurst fractal interpolation is a valid possibility for restoring data in a surface in such a way that maintains the scaling relations. Hurst interpolation would only be useful for restoring data in the lateral (x) direction, the downstream (y) direction has no discernible scaling relation. A cubic spline would suffice to fill in missing data in the y-direction.

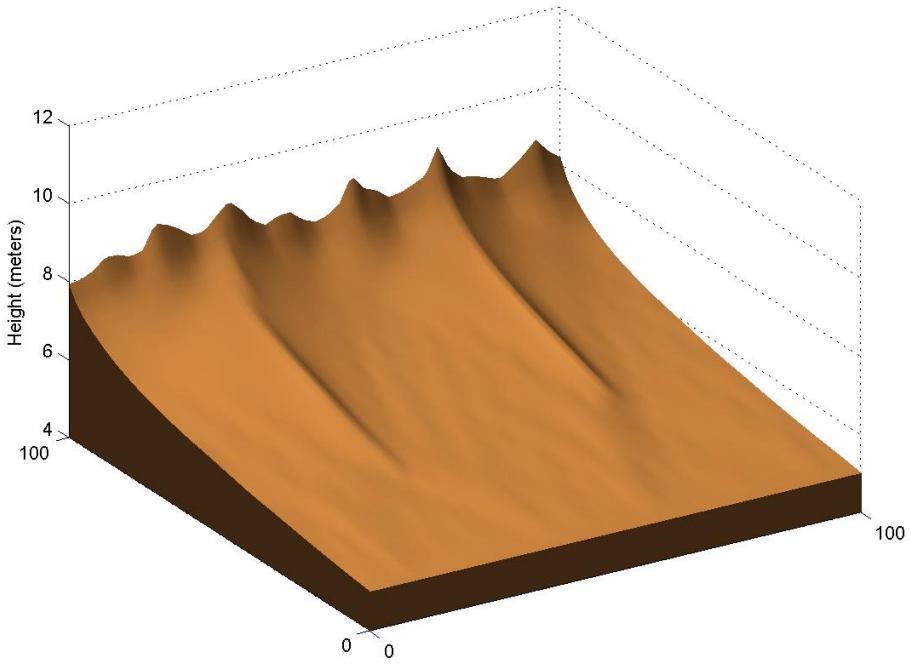


Figure 5.4: A surface generated on a coarse grid of 41 by 41 grid points

Figure 5.4 shows a surface that was generated on a coarse gird of 41 by 41 grid points. This surface took a couple of weeks to erode, but it has only two of the expected three mountain ridges. While interpolation techniques can be used to transpose this data onto a finer mesh, it will not be able to add the missing mountain ridge. This multigrid technique would be better suited for making a finer discretization from the typical discretization of one square meter grid cells.

5.2 Artificial Construction

For a hundred by one hundred square meter area, the statistical theory mentioned previously implies that a resulting eroded surfaces will contain three mountain ridges separating three river valleys. The Brownian motion of water over the surface during the channelization phase is what forms these mountain ridges and causes them to meander down the ridge. A history of tests imply that the free water surface at 60% eroded has a variogram scaling of around 0.75, and Hack's Law provides scaling laws for the gradient of the water surface $|\nabla H|$ [12, 10]. The goal is to use these facts to artificially generate an eroded surface. Here, the phrase "artificially generate" or an "artificial surface" implies the generation of a numerical solution of Eq. (2.6) without the use of any numerical scheme developed to solve the Smith and Bretherton model.

The general formula for an artificial surface (around 60% eroded) is

$$H(x,y) = B(y) + \sum_{i=0}^{N} R_i(x,y) - \sum_{k=0}^{M} V_k(x,y) \tag{5.1}$$

where $B(y)$ is the minimum height of the surface a distance y from the base of the slope, $R_i(x,y)$ is a function for the i^{th} mountain ridge, and the $V_k(x,y)$ are valleys cut out of

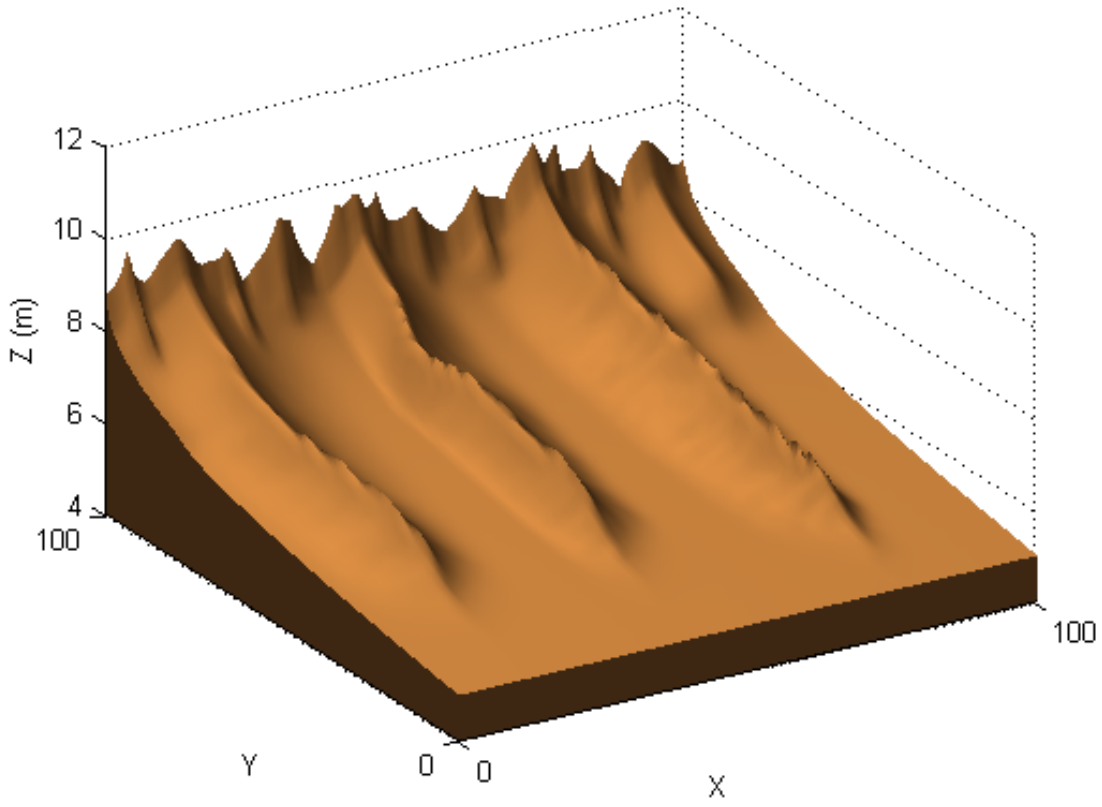


Figure 5.5: An artificially generated surface

the surface for extra realism. Figure 5.5 is an example of a surface generated from this equation. The general form for all these functions and how they were found is given below.

5.2.1 The Ridge Function

Separable solutions to Eqs. (2.5) and (2.6) where studied by Birnir and Rowlett [11]. They found that

$$H_0(x,y) = \Big(H_1^{1/c} + a(x-x_0) + b(y-y_0)\Big)^c \tag{5.2}$$

constitutes mountains and mountain ridges for appropriate constants a, b, c, x_0, y_0, and H_1. For mountain ridges specifically, a and b must have opposite sign with the sign changing across $x - x_0 = y - y_0$.

This equation has a number of complications including finding appropriate values for all the constants as well as having to be vigilant regarding the domain of the function. This equation also models a mountain ridge of infinite extent (in the downstream direction) and not a collapsing ridge.

For these reasons, a simpler model for the mountain ridges was designed. These mountain ridges consist of two functions: one that defines its height and one that defines its width.

For the lateral direction (width) the function

$$W(x) = e^{-|x|}, \tag{5.3}$$

was chosen. The first two terms in the infinite series expansion are reminiscent of Eq. (5.2), and this function matches the general shape of mountain ridges found in simulations and by Birnir and Rowlett [11]. One benefit of this function is its maximum height of one, making it simple to have a secondary function to control its height. The primary benefit of this function is that it (practically) has compact support, making it ideal to add multiple translated copies of this function together for the various mountain ridges.

To find a suitable height function multiple simulated surfaces were examined. Figure 5.6 shows the data for one surface: the maximum height of each mountain ridge plotted per lateral cross section. Each mountain ridge remains relatively flat before entering an initial growth period. These ridges find a stable height above the valley basin before one final jump in height at the top boundary. Ignoring this last burst in height, the general

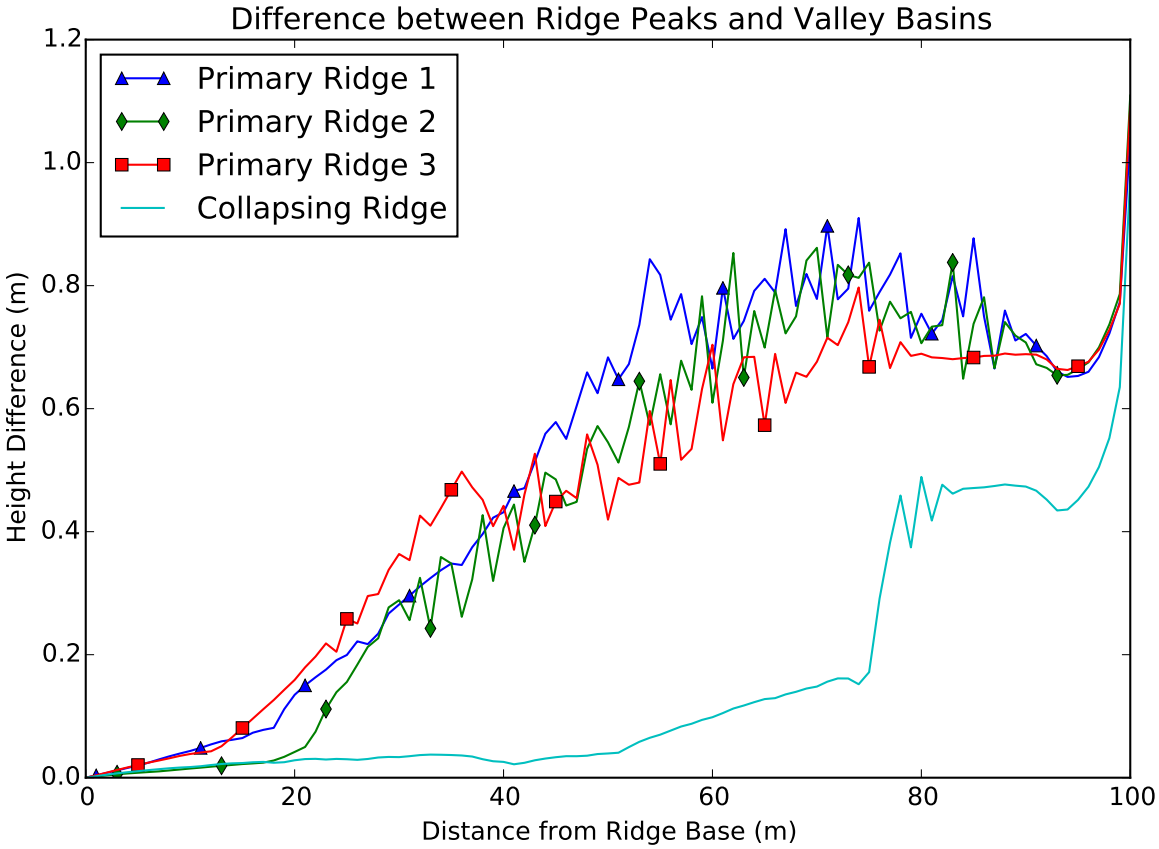


Figure 5.6: Difference between ridge peaks and valley basins for a typical surface

shape is best captured by the logistic function

$$H(y) = \frac{1}{1 + e^{-y}}. \tag{5.4}$$

Combining these equations together yields the general mountain ridge equation

$$R_i(x, y) = \frac{m_i}{1 + 100 \exp\left(-r_i \cdot (y - s_i)\right)} \cdot \exp\left(-w_i \cdot |x - p_i|\right) \tag{5.5}$$

where $m_i, r_i, s_i, w_i,$ and p_i are all positive constants. Specifically m_i is the maximum height of the ridge above the valley basin, r_i controls the rate at which the ridge grows

to its maximum height, s_i is where the ridge starts to emerge, w_i controls the maximum width of the ridge in the lateral direction, and p_i is the location of the ridge peak.

The constant of 100 in Eq. (5.5) is not a completely arbitrary constant. The goal of the s_i term is to shift the logistic function such that $y = s_i$ is about when the mountain ridge begins emerging. The horizontal shift $y \mapsto y - s_i$ into Eq. (5.4) does not achieve this goal, but instead shifts the graph such that the translation takes the value $1/2$ at $y = s_i$. The constant of 100 is essentially a permanent horizontal shift that allows the s_i term to have the desired property in the function.

Equation (5.5) is used for all the mountain ridges, including the short-lived ones near the top of the slope. In general, the closer to this boundary a mountain ridge emerges (the larger s_i is), the quicker it increases in height (larger r_i value), and the narrower the ridge (larger w_i value). The values of these constants are all generated at random when a mountain ridge is added to the surface. The range of potential values for these constants was found by observation and comparisons with model generated surfaces. Section 5.2.3 describes how the locations of these mountain ridges are determined.

The three primary mountain ridges have one particular characteristic that makes their generation different from the others. These mountain ridges meander down the slope, and as a result, have more jagged peaks. To simulate this, randomness is added to the ridge function in the form of Brownian motion. The current theory suggests that the meandering of these mountain ridges is caused by the Brownian motion of water movement down the initial surface [12, 10]. This Brownian motion is added to the x value when the value of $R_i(x, y)$ is calculated. This randomness is only added to the x value as (i) this is what causes the shift of the mountain peak laterally along the surface and (ii) in early simulations there was no noticeable change in the surface when randomness was also added to the y component.

5.2.2 The Valley Function

At the top boundary, the many mountain peaks and valleys form a jagged profile. Due to the choice of width functions, their summation creates a much smoother profile than what occurs in simulations, see Fig. 5.7.

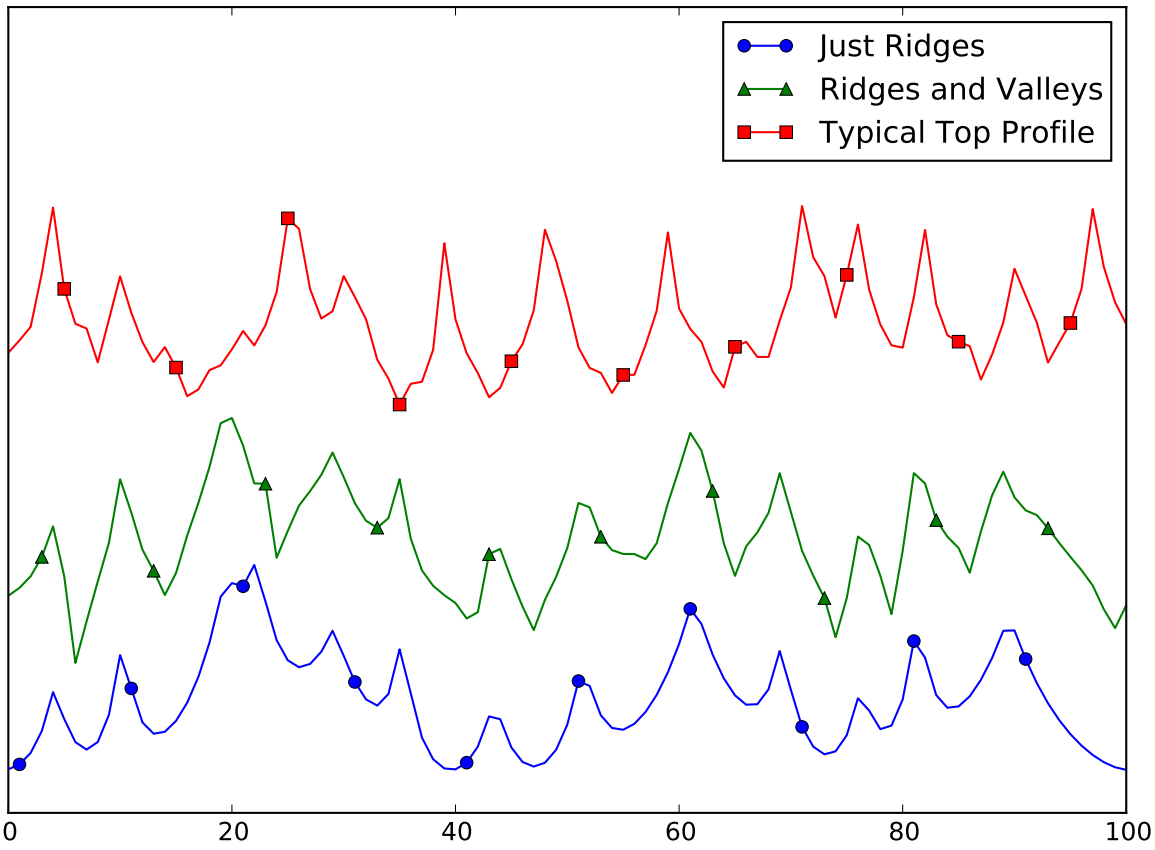


Figure 5.7: Profiles of a typical surface, and an artificial surface with and without valleys

It is only near the top boundary where something extra is needed to better define the ridges and valleys, as many of the smaller mountain ridges quickly erode away into one of the three primary river valleys. The crude observation was made that these valleys

vaguely resemble a mountain ridge turned upside-down. Thus,

$$V_k(x,y) = \frac{m_k}{1 + 100\exp\left(-r_k \cdot (y - s_k)\right)} \cdot \exp\left(-w_k \cdot |x - p_k|\right) \tag{5.6}$$

is used for these valleys, for suitable constants m_k, r_k, s_k, w_k, and p_k.

5.2.3 Feature Selection

The functions for the ridges and valleys require knowing were those features are located laterally on the surface. Birnir et al. [10] found a formula for the expected number of valleys at the top of the slope. Considering the periodic boundary conditions, this formula also dictates the number of mountain ridges, and thus is used to find N (and M). The resulting value of N is typically not an integer, and so it gets rounded up or down randomly each time a new surface is created.

Next, the lateral direction is divided in N pieces of equal length. Within each of these pieces, a point is selected from a normal distribution to be a mountain peak. The mean of this distribution is the center of the interval, and the variance is one-quarter the interval's length. This method distributes mountain peaks relatively evenly in the lateral direction as occurs in many simulations. This method helps prevent any two ridges from forming practically on top of one another, which may happen when using uniform distributions. There are positions laterally that will almost never contain a mountain peak with this method (e.g., the boundary between two sections). This is solved with a simple horizontal shift of the entire surface as the boundary conditions in this direction are periodic.

Once all the mountain peak locations have been selected, valley locations are then chosen. Between each pair of adjacent peaks a valley is placed. The location of a valley is also taken from a normal distribution, with mean in the center of the peaks and variance

of one-fifth the distance separating them. The normal distribution is again used to pick a placement randomly, but still somewhere near the center of the ridges to prevent overlapping a peak with a valley.

Last is to determine which of these peaks represent the primary mountain ridges, which is the collapsing ridge, and which are not. Like with the selection of the peak locations, there should be a respectable amount of separation between the primary ridges. When a peak location is selected as a primary ridge, the algorithm is then prevented from selecting the four nearest peaks as a primary ridge. Once the three primary ridges are selected, the collapsing ridge is chosen with the constraint that it can not be adjacent to any of the primary ridges.

In some simulations a second unstable, collapsing ridge is present at the 60% eroded mark. The next mountain peak is selected with the same constraint of not being adjacent to any of the previously selected peaks. There is a small probability, around 25%, that this peak will also be a collapsing ridge when the surface is generated.

The remaining ridges are going to be the short-lived ridges near the top boundary. The ordering of the mountain peaks is how the algorithm knows which peaks are the primary ridges, the unstable collapsing ridges, etc. These ridges have no constraint on their placement with respect to the primary features, and thus are randomly ordered.

5.2.4 The Base Function

The mountain peak height in the ridge function is based on finding the height above the valley basin. To find a function for the valley basin, again, a number of surfaces were examined in search for some kind of pattern/elementary function that may interpolate the data.

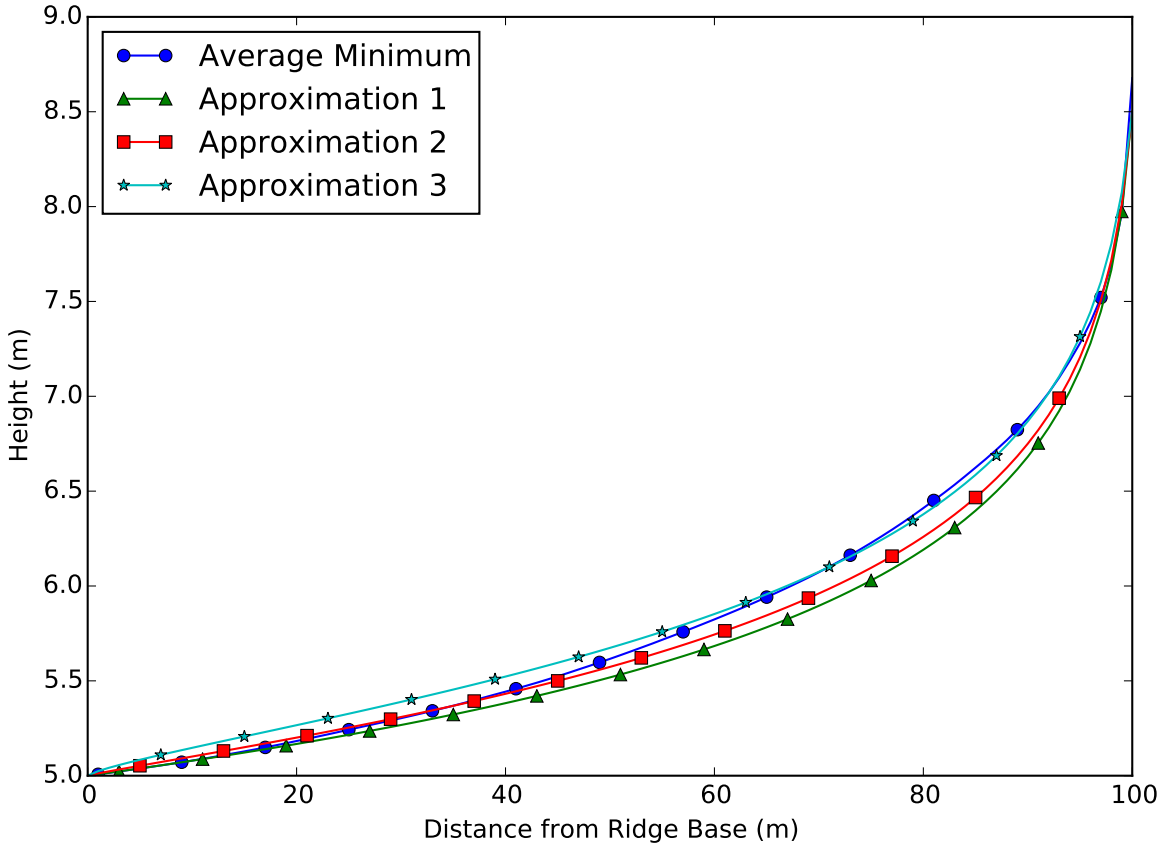


Figure 5.8: Average minimum height of surfaces with possible approximation functions

There was no one best function, but a family of functions given by

$$B(y) = (C + H_0) - C \cdot \log_{101}\Big(101 - f(y)\Big) \tag{5.7}$$

where H_0 is the lower boundary condition of Eq. (2.10), the quantity $C + H_0$ is the desired (minimum) height at the top of the slope, and $f(y)$ is a continuous satisfying $f(0) = 0$ and $f(100) = 100$. All three approximation functions in Fig. 5.8 have $H_0 = 5$ and $C = 3.5$. The first approximation has $f(y) = y$, the second is $f(y) = 100^{\frac{1}{10}} y^{\frac{9}{10}}$, and the third is $f(y) = 100^{\frac{1}{4}} y^{\frac{3}{4}}$.

5.2.5 Small Scales

Surfaces generated by the Smith and Bretherton model are known to contain small scale features that are best captured by the variogram [12, 10]. The functions and general construction described above generate the necessary large scale features, but a priori, are not guaranteed to contain the small scales. Three methods, all making use of Hurst fractal interpolation, are used in an attempt to add the small scale dynamics to these surfaces. The effectiveness of these methods are examined in Sect. 5.3.2.

Lateral Hurst Interpolation

This is the method as described in Sect. 4.3 for the initial tests of restoring data using Hurst fractal interpolation. Again, both the extrema and DP methods are used to select the feature points of the surface laterally, which then get used as the interpolation points.

Piecewise Hurst Interpolation

The main downfall of the lateral Hurst interpolation method is that it adds a series smaller peaks into the valleys, see Fig. 5.9. The proposed solution to this is to not interpolate the entire cross section at once, but to do so in a piecewise manner.

For this technique, both the extrema and DP methods are used to break down a cross section into linear pieces. For each adjacent pair of feature points, a couple more points are selected between them. The Hurst interpolation is then performed on that subset of points. This method requires more points and multiple different Hurst interpolations per cross section, but interpolating in these smaller chunks removes the "mountain bias" incurred from the lateral Hurst interpolation method.

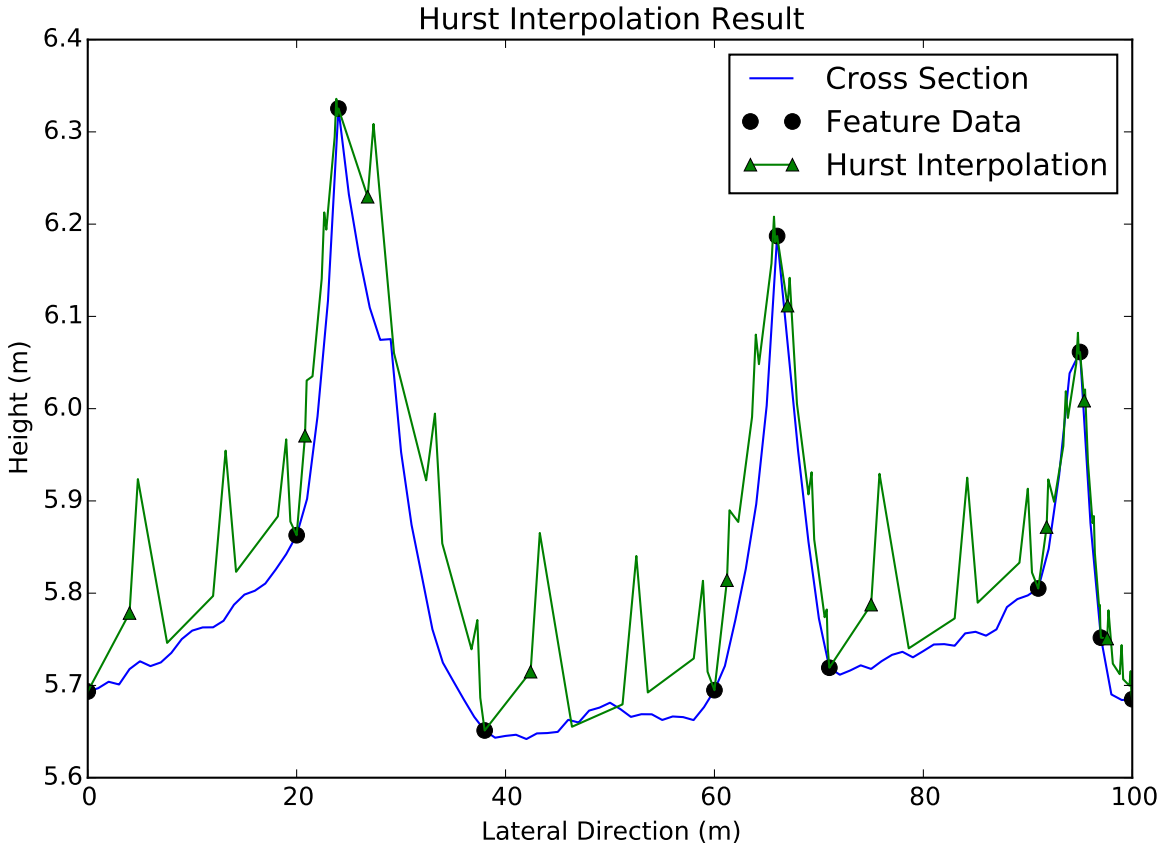


Figure 5.9: A cross section, its feature points, and the resulting Hurst interpolation

Independent Noise

The Hurst interpolation methods described above generate small scale dynamics from a pre-existing surface. There is no reason, however, to assume that there is a relationship between the large and small scales, other than how the smoothness of the large scales skew the variogram of the free water surface, see Sect. 2.3. To test this, small scale noise was generated independently of the large scale features, then added to the surface. As with the other methods of adding the small scale features, two slightly different methods were used.

Both methods involve creating a noisy surface similar to fractal waves. In one direction, the direction perpendicular to the base of the slope, cosine curves are used to generate a series of initial points. The amplitude of these waves is altered to assure that the resulting noise would not dominate the general shape of the landscape when added to the large scale features. In the lateral direction Hurst interpolation is used to fill in the rest of surface. The first method adds this resulting noisy surface directly to the large scales.

The second method integrates the noisy surface in the y (perpendicular) direction before adding it to the large scale surface. The reason is due to the realization in Sect. 2.3 that the better scaling results are from the gradient of the surface, and not the surface itself. The integration of the noise is an attempt to add the small scale dynamics to the gradient of the artificial surface instead of the surface itself.

5.3 Results and Accuracy

Sections 5.1 and 5.2 present a number of different ways to accelerate the process of fluvial erosion. The success of these methods to approximate the desired numerical results is measured via three criteria: (i) does the surface contain the large scale features, (ii) does the surface contain the small scale features, and (iii) can the surface be eroded by the current code implementation?

5.3.1 Numeric Modification Results

All the surfaces generated by the methods described in Sect. 5.1 obviously satisfy the third condition. Of the surfaces generated using a tolerance of 10^{-6}, only about 10%

of them satisfied the first condition. For the computing of the small scale dynamics, it is desired to only use the surfaces that maintained the desired large scale features. Unfortunately, too few of those surfaces successfully developed to get accurate scaling results, and so by default, these surfaces do not satisfy the second condition. Therefore the use of a tolerance of 10^{-6} is too high to get reliably accurate surfaces from this model.

Surfaces generated on coarser grids also failed to have the necessary large scale features. Due to how poorly these surfaces captured the large scale dynamics there is no reason to calculate the small scale features. This method of accelerating results is also too inaccurate to be useful.

Surfaces generated by alternating the tolerance did all generate the expected number of mountains, mountain ridges, and valleys. If there is any criticism over these surfaces, it is that the primary ridges that form are slightly short, but this is expected considering the surfaces generated with the constant high tolerance. Figure 5.10 shows the small scale results of these surfaces.

To reiterate, the scaling results are most accurate in the center. The scaling for the free water surface H is expected to hover around 0.75 while the scaling for the gradient should be around 0.25 [12, 10].

Thus surfaces generated by alternating between tolerances capture all the desired scaling results. It is expected that using a lower tolerance more often will cause both the large scale mountain ridges to better assert themselves and generate more consistent small scale results, at the sacrifice of computation time. Thus the switching between tolerance is a valid method for more quickly generating these surfaces.

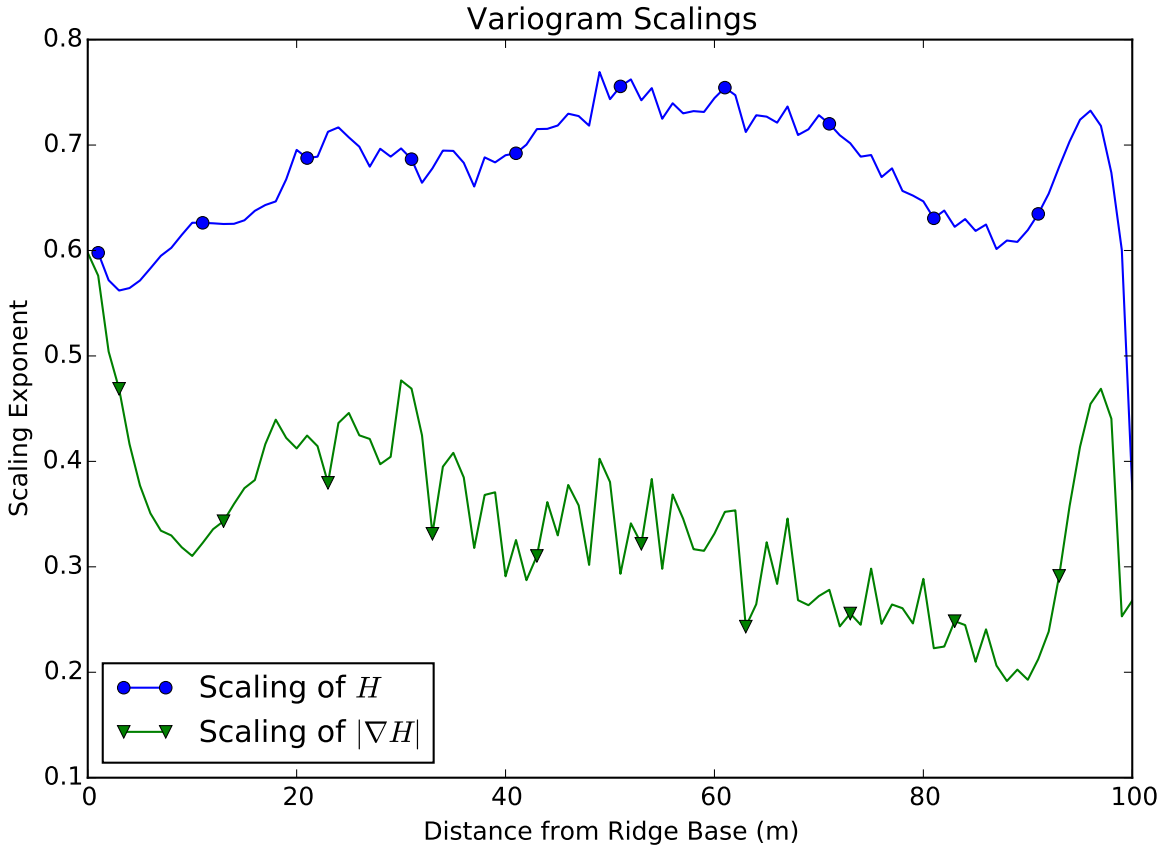


Figure 5.10: Variogram scalings for surfaces generated by switching the tolerance

5.3.2 Artificial Construction Results

The code for the artificial construction was designed so that the resulting surfaces would satisfy the large scale dynamics. Multiple different techniques are described in Sect. 5.2.5 for adding in the desired small scale features, and the results of these techniques are given below. Figure 5.11 shows the variogram scalings for all the methods discussed in Sect. 5.2.5, while Fig. 5.12 shows the scalings of the gradients of those surfaces.

For these variograms, a number of surfaces were generated without any addition of small scales features. These surfaces are referred to as "basic" surfaces from here on. Then each basic surface had the small scale features added in each of the six methods discussed

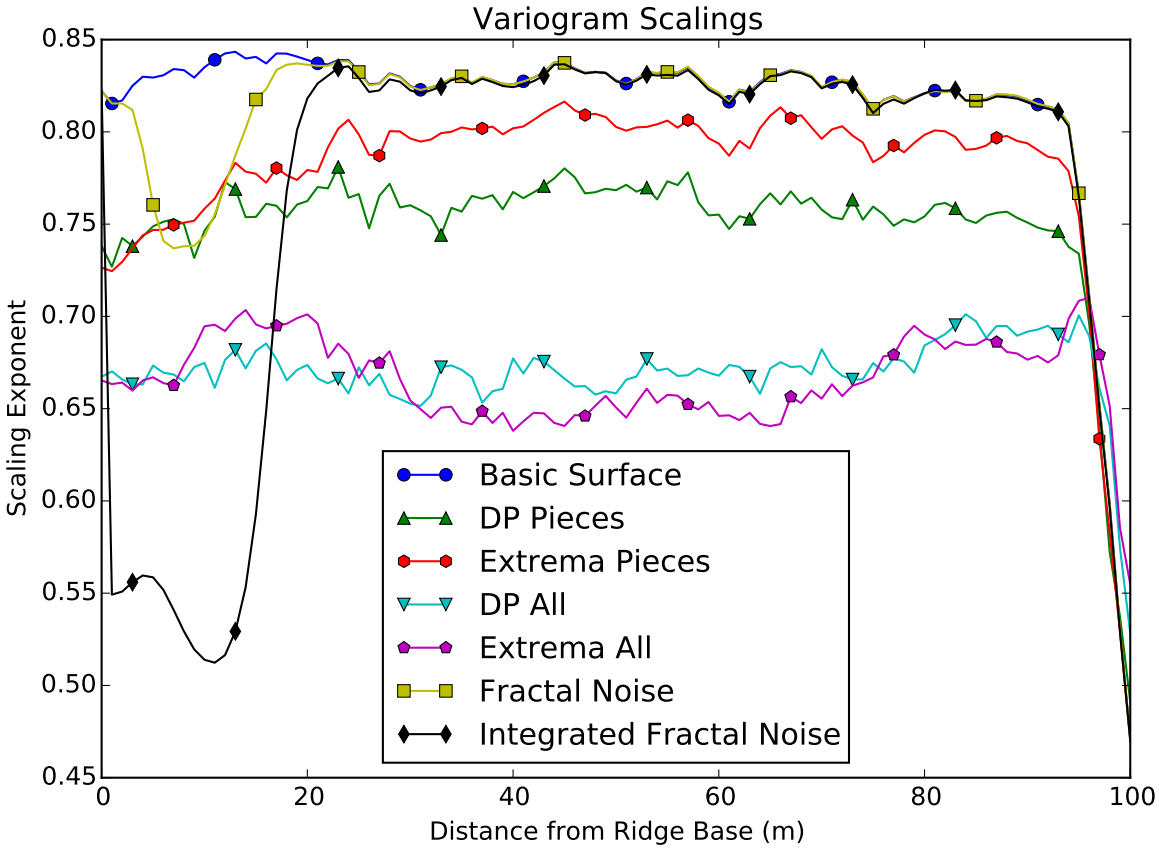


Figure 5.11: Variogram scalings for artificially generated surfaces

above. This allows for a more accurate depiction of how each of these methods of adding the small scales affect the statistics.

In the legends, "DP" or "Extrema" refer to the algorithm used for finding the feature points of a cross section. The "All" refers to the use of lateral Hurst interpolation while "Pieces" is the piecewise interpolation. There are many points of interest within in these variograms, which are discussed below.

The first is the scaling of the basic surfaces, whose variogram hovers rather consistently between 0.80 and 0.85. These values, while higher than the expected, still fall in a range seen by other simulations [65, 57].

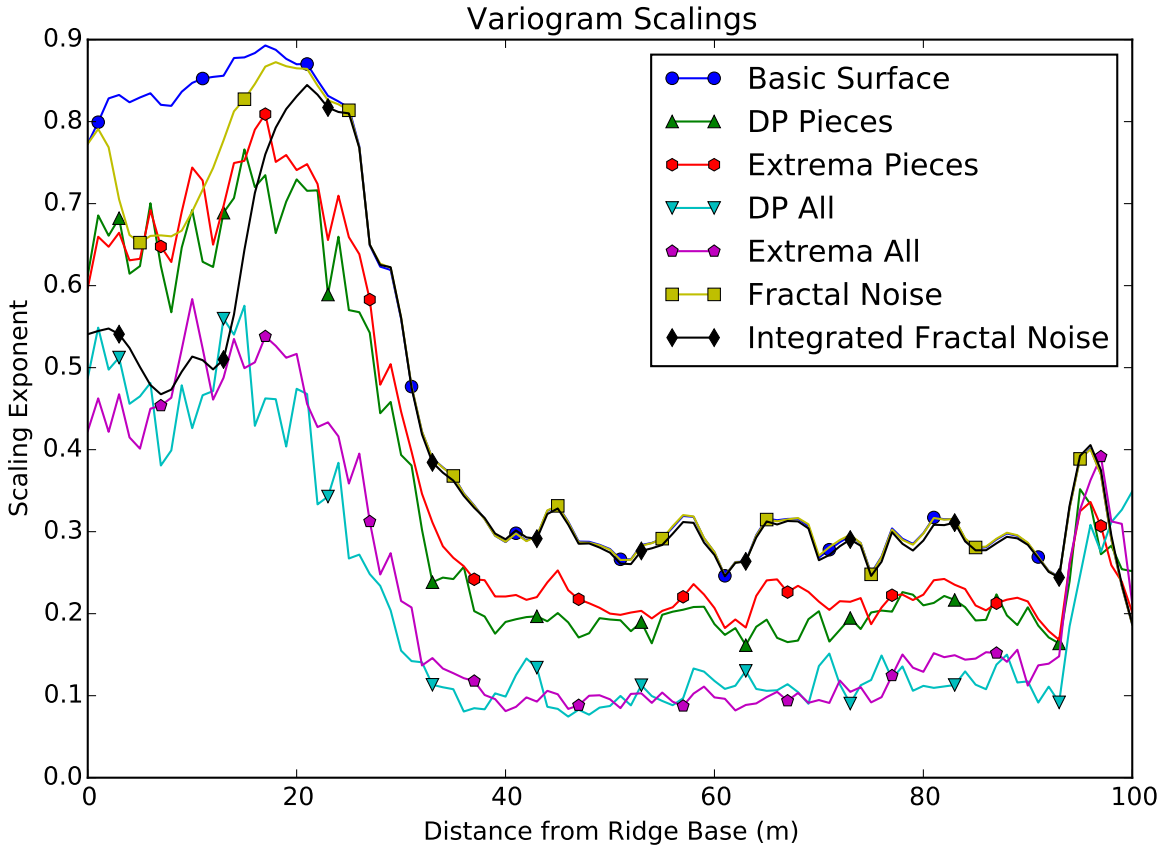


Figure 5.12: Variogram scalings for the gradients of artificially generated surfaces

The scalings of the basic surface gradients present a number of interesting results. Initially the scaling exponent transitions from 0.8 up to 0.9, both unusually high values, especially for the gradient scaling. In this case, however, these values make sense as the generation code alters very little of the surface within the first 20 meters from the bottom of the slope. Thus the gradient there should be very close to a flat plane. Furthermore, the scaling of the variogram is the Hurst exponent, which is a measure of smoothness, with 0 being very rough and 1 being smooth (straight lines for example have a Hurst exponent of 1). Thus the gradient is very smooth near the base of these surfaces.

The transition to a lower scaling (more rough) makes sense, but the fact that exponent then hovers around 0.30 is particularly interesting. The roughness exponent for gradient has been calculated to be 0.70 [12], which means the Hurst exponent should be 0.30 (recall Sect. 2.3). Thus these basic surfaces capture the small scale dynamics over majority of the surface without outside help.

The only noticeable effect either fractal noise technique has on the scalings is near the base of the slope. Again this makes sense as this region is relatively flat, so the added small scales dominate the variogram. Once the mountain ranges are formed, their large scales then dominated the small scales introduced by the fractal noise.

The lateral Hurst interpolation has the same scaling results as seen in Sect. 4.3 in the initial tests. The extreme roughness of these surfaces is further supported by the gradient scaling, which is the lowest (most rough) of all the methods for adding small scales.

Performing the Hurst interpolation in pieces was the most successful method. Both scalings in the surface are closest to the 0.75 scaling used for the Hurst interpolation, with the DP method of point selection perfectly hovering at that scaling. The gradient scalings are in an acceptable range for these surfaces and match early results of the Birnir et al. [12], see Sect. 2.3.

For the third criterion, a series of surfaces were run through the code. Five basic surfaces were selected along with their piecewise DP interpolated, lateral extrema interpolated, and additional fractal noise counterparts for testing. This mix has one representative from each small scale addition method as well a DP selection and an extrema selection method. Since these methods have similar scaling results and similar selection points (Fig. 4.3), it is likely that they will behave in the same manner when run through the numerical methods.

One flaw that the Hurst interpolation methods have is that the scalings added in the lateral directions are independent of one another. This can cause natural dams to form that impede the solving of Eq. (2.5) numerically. In these cases, it has been observed that the water depth at these points increases (seemly) without bound. This phenomenon has been observed at no more than ten locations in early tests.

To counter this, the code was altered so that equilibrium occurs when majority (at least 90%) of points reach an equilibrium water depth, instead of requiring equilibrium at all points. This is not seen as a violation of condition (iii) as these natural dams/reservoirs occur in nature and/or the drainage direction may be in a diagonal direction, but the upwind scheme is incapable of accounting for that kind of formation. Usually, after quarter of a percent of erosion, these features have been eroded away by the code and waiting for an equilibrium at every point can resume.

Assuming an initial linear ridge with a 5% grade, the generated surfaces lie in the 50% to 58% eroded range. The goal is not only to have these surfaces run in the code, but erode until they reach the 60% mark and then investigate the variogram data. All five of the basic surfaces, piecewise DP interpolated surfaces, and the surfaces with fractal noise continued the erosion process via the code. The variograms of the resulting eroded surfaces can be found in Fig. 5.13 and the gradient scalings are in Fig. 5.14. Sample artificially generated surfaces, as well as some of the resulting surfaces after running through the numerical methods can be found in Appendix B.

The most unexpected result is the similarity between the variograms. It was believed that the various methods of adding small scale features would alter the flow of water over the surface, which would intern alter the scalings of the eroded surfaces more distinctly. While the scalings are diverse, the differences are much more subtle. It appears that the large scale features have more of a control over the small scales than initially thought.

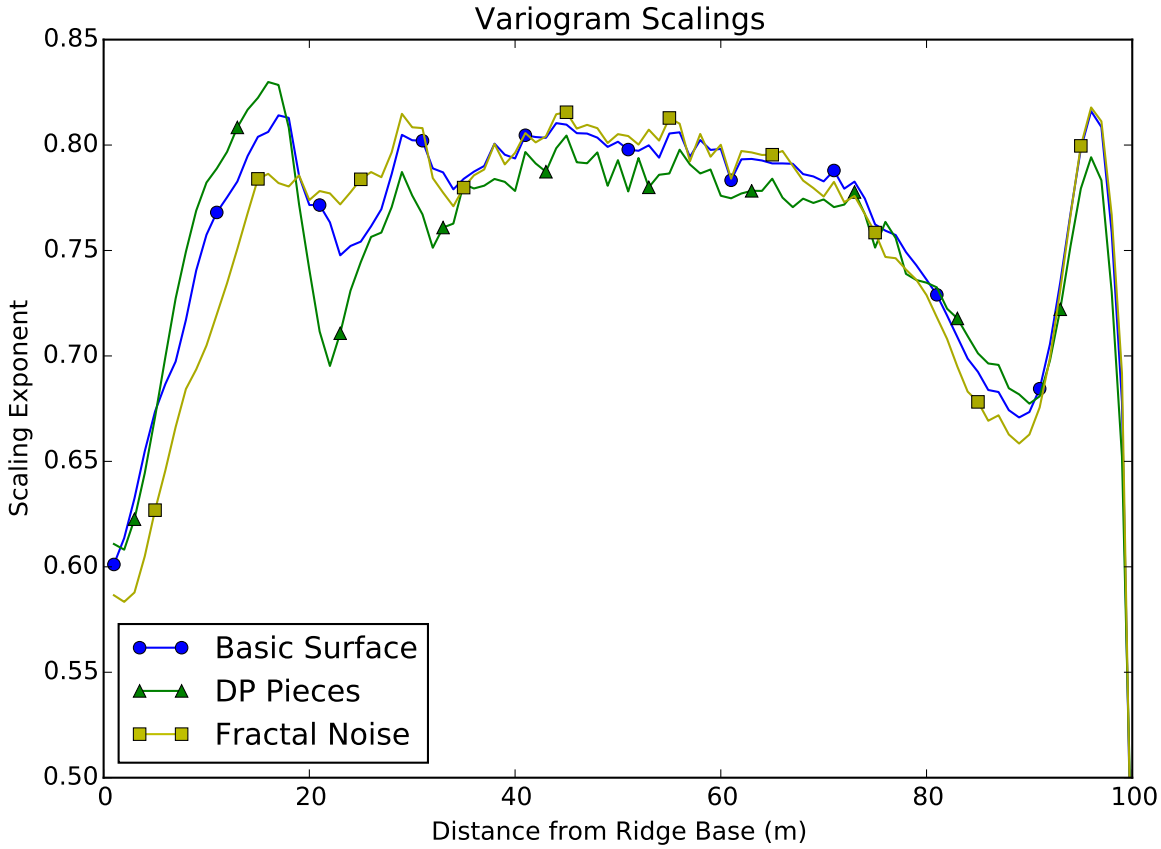


Figure 5.13: Artificial surface variograms after running in the code

Otherwise, the variogram results are ideal. The scaling results for the surfaces hover between 0.75 and 0.80 for the majority of the variogram. A quick comparison between Figs. 5.11 and 5.13 shows how the boundary conditions at the top and bottom of the slope skew the scaling results. The gradient scaling results hover nicely between 0.25 and 0.30, exactly as expected for these surfaces [12]. Again, the skewing of the variogram due to the boundary conditions is clearly visible in the gradient scaling as well. The cause of the large jump in scaling between the 20 and 30 mark is still not fully understood. It appears to be a remnant of the initial transition of scalings seen in Fig. 5.12 that has not yet been fully eroded by the code.

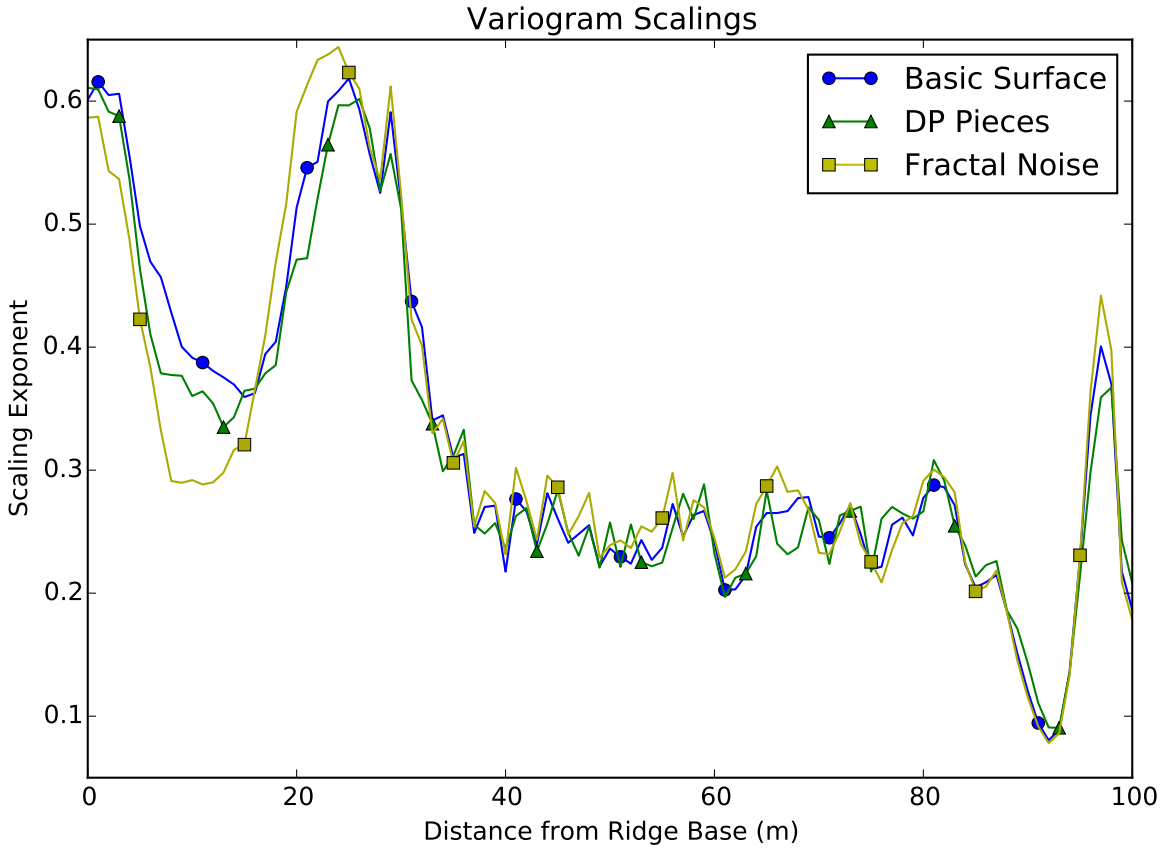


Figure 5.14: Artificial surface gradient variograms after running in the code

Based on these results, surfaces generated by Eq. (5.1) satisfy the three criteria for quick generation of an artificial solution to the Smith and Bretherton model. Most surprising is the fact that the basic surfaces generated by this equation satisfy the small scale dynamics. The use of fractal noise or piecewise Hurst interpolation can be used for additional forcing of these small scales, if desired.

After three weeks of computation, none of the five surfaces with lateral Hurst interpolated roughness have eroded via the code. This is sufficient time without result to say that these surfaces are too rough for the current code implementation and thus fail the third condition.

Chapter 6

Conclusions

Of the numerical methods used to solve the Smith and Bretherton model, the MacCormack scheme requires the least programming, but correctly adding the artificial viscosity is an extremely tedious process. Moreover, even with an appropriate amount of artificial viscosity, this method still fails to capture the crucial small scale dynamics related to Hack's Law.

The upwind/Crank-Nicolson scheme, although more difficult to program, is simpler to implement as extra viscosity does not need to be added to the scheme. More important is the fact that the Crank-Nicolson scheme captures the small scale features of the landscape. These methods, both optimized for calculating overland flow, reveal that implicit schemes are best equipped for capturing crucial small and large scale dynamics.

In either case, the numerics are too slow to be of practical use, hence the need to find ways to accelerate the process. Of the methods that modify the code itself, the switching between tolerance values is the most promising method. More work into this method will greatly accelerate the generation of eroded surfaces with little to no loss of accuracy. The

success of this technique brings into question more adaptive methods, such as changing the tolerance on a point by point basis, or the possibility of not continuing to solve for the water depth at points that have already achieved equilibrium.

The artificial construction of surfaces and the methods of adding the small scale features have answered, and created, questions about the statistical theory. Hack's Law implies that the roughness exponent of these surfaces should be $\chi \approx 0.75$ which further implies that the Hurst exponent is $1 - \chi \approx 0.25$. The variogram scalings, which is a measure of the Hurst exponent, of the basic surfaces confirms the fact that the topography of the landscape smooths the surface, which increases the calculated Hurst exponent. Therefore the most accurate scalings of the variogram come from $|\nabla H|$, the gradient of the free water surface.

The two different Hurst interpolation methods for adding small scales accentuate the separation of the large and small scales of these surfaces. The lateral Hurst interpolation method forces the large scale dynamics into the small scales, which is what causes the peaks in the river valleys and the lower scaling (Hurst) exponent of these surfaces. The piecewise interpolation does not have this mountain bias, but instead alters the already existing small scale dynamics to the desired Hurst exponent. Piecewise Hurst interpolation, when using the Douglas-Peucker method to select feature points, is the best method for altering a landscape and forcing it to contain the desired small scales. This method also shows that forcing the (Hurst exponent) scaling of the surface H to 0.75 forces a scaling of 0.25 on the gradient, but exactly how or why this works is to be determined.

Based on the scaling results in Figs. 5.13 and 5.14, however, the large scale features have a strong influence over the resulting small scales. Exactly what this influence is or how it acts on the small scales is still to be determined.

Bibliography

[1] Barnsley, M. *Fractals Everywhere*. Dover Books on Mathematics. Dover Publications, 2012.

[2] Barnsley, M. F. Fractal functions and interpolation. *Constructive approximation 2*, 1 (1986), 303–329.

[3] Barnsley, M. F., and Demko, S. Iterated function systems and the global construction of fractals. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 399*, 1817 (1985), 243–275.

[4] Barnsley, M. F., Devaney, R. L., Mandelbrot, B. B., Peitgen, H.-O., Saupe, D., Voss, R. F., Fisher, Y., and McGuire, M. *The science of fractal images*. Springer Publishing Company, Incorporated, 2011.

[5] Barnsley, M. F., Elton, J., Hardin, D., and Massopust, P. Hidden variable fractal interpolation functions. *SIAM Journal on Mathematical Analysis 20*, 5 (1989), 1218–1242.

[6] Barnsley, M. F., and Harrington, A. N. The calculus of fractal interpolation functions. *Journal of Approximation Theory 57*, 1 (1989), 14–34.

[7] Bedford, T. Hölder exponents and box dimension for self-affine fractal functions. *Constructive Approximation 5*, 1 (1989), 33–48.

[8] Bierman, R. B., and Montgomery, D. R. *Key Concepts in Geomorphology*. W. H. Freeman and Company Publishers, New York, N.Y., 2014.

[9] Birnir, B. Turbulent rivers. *Quarterly of Applied Mathematics 66* (2008), 565–594.

[10] Birnir, B., Hernández, J., and Smith, T. R. The stochastic theory of fluvial landsurfaces. *Journal of Nonlinear Science 17*, 1 (2007), 13–57.

[11] Birnir, B., and Rowlett, J. Mathematical models for erosion and the optimal transportation of sediment. *International Journal of Nonlinear Sciences and Numerical Simulation 14* (2013), 232–337.

[12] Birnir, B., Smith, T. R., and Merchant, G. E. The scaling of fluvial landscapes. *Computers & Geosciences 27*, 10 (2001), 1189 – 1216.

[13] Chand, A., and Kapoor, G. Generalized cubic spline fractal interpolation functions. *SIAM Journal on Numerical Analysis 44*, 2 (2006), 655–676.

[14] Chen, C.-J., Lee, T.-Y., Huang, Y., and Lai, F.-J. Extraction of characteristic points and its fractal reconstruction for terrain profile data. *Chaos, Solitons & Fractals 39*, 4 (2009), 1732 – 1743.

[15] Dingman, S. L. *Fluvial hydrology*. WH Freeman and Company, New York, NY, 1984.

[16] Douglas, D. H., and Peucker, T. K. *Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature*. John Wiley & Sons, Ltd, 2011, pp. 15–28.

[17] Edwards, S. F., and Wilkinson, D. The surface statistics of a granular aggregate. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* (1982), vol. 381, The Royal Society, pp. 17–31.

[18] Falconer, K. *Fractal Geometry: Mathematical Foundations and Applications*. Wiley, 2014.

[19] Fennema, R. J., and Chaudhry, M. H. Explicit numerical schemes for unsteady free-surface flows with shocks. *Water Resources Research 22*, 13 (1986), 1923–1930.

[20] Fernández-Martínez, M., and Snchez-Granero, M. A new fractal dimension for curves based on fractal structures. *Topology and its Applications 203* (2016), 108 – 124. The Iberoamerican Conference on Topology and its Applications 2014.

[21] Fisher, Y. *Fractal image compression: theory and application*. Springer Science & Business Media, 2012.

[22] Fournier, A., Fussell, D., and Carpenter, L. Computer rendering of stochastic models. *Commun. ACM 25*, 6 (jun 1982), 371–384.

[23] Fowler, A. C., Kopteva, N., and Oakley, C. The formation of river channels. *SIAM Journal on Applied Mathematics 67*, 4 (2007), 1016–1040.

[24] Génevaux, J.-D., Galin, É., Guérin, E., Peytavie, A., and Benes, B. Terrain generation using procedural models based on hydrology. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 143.

[25] Hack, J. T. Studies of longitudinal stream profiles in virginia and maryland. *U.S. Geological Survey Professional Paper 294-B* (1957).

[26] Hardin, D., and Massopust, P. R. Fractal interpolation functions from $\mathbb{R}^n$ into $\mathbb{R}^m$ and their projections. *Zeitschrift für Analysis und ihre Anwendungen 12*, 3 (1993), 535–548.

[27] Hnaidi, H., Guérin, E., Akkouche, S., Peytavie, A., and Galin, E. Feature based terrain generation using diffusion equation. In *Computer Graphics Forum* (2010), vol. 29, Wiley Online Library, pp. 2179–2186.

[28] Horton, R. E. Erosional development of streams and their drainage basins; hydrophysical approach to quantitative morphology. *Geological Society of America Bulletin 56*, 3 (1945), 275–370.

[29] Howard, A. D. Theoretical model of optimal drainage networks. *Water Resources Research 26*, 9 (1990), 2107–2117.

[30] Howard, A. D. A detachment-limited model of drainage basin evolution. *Water Resources Research 30*, 7 (1994), 2261–2285.

[31] Hsü, K. J., and Hsü, A. J. Fractal geometry of music. *Proceedings of the National Academy of Sciences 87*, 3 (1990), 938–941.

[32] Huang, Y., and Chen, C.-J. 3d fractal reconstruction of terrain profile data based on digital elevation model. *Chaos, Solitons & Fractals 40*, 4 (2009), 1741 – 1749.

[33] Ikeda, S., Parker, G., and Sawai, K. Bend theory of river meanders. Part 1. Linear development. *Journal of Fluid Mechanics 112* (Nov. 1981), 363–377.

[34] Izumi, N. The formation of submarine gullies by turbidity currents. *Journal of Geophysical Research: Oceans (1978–2012) 109*, C3 (2004).

[35] Izumi, N., and Fujii, K. Channelization on plateaus composed of weakly cohesive fine sediment. *Journal of Geophysical Research: Earth Surface (2003–2012) 111*, F1 (2006).

[36] Izumi, N., and Parker, G. Inception of channelization and drainage basin formation: upstream-driven theory. *Journal of Fluid Mechanics 283* (1995), 341–363.

[37] Izumi, N., and Parker, G. On incipient channels formed at the downstream end of plateaux (in Japanese). *J. Hydraul. Coastal Environ. Eng. JSCE*, 521 (1995), 79–91.

[38] Izumi, N., and Parker, G. Linear stability analysis of channel inception: downstream-driven theory. *Journal of Fluid Mechanics 419* (2000), 239–262.

[39] Jovanović, M., and Djordjević, D. Experimental verification of the maccormack numerical scheme. *Advances in Engineering Software 23*, 1 (1995), 61–67.

[40] Julien, P., and Simons, D. Sediment transport capacity of overland flow. *Transactions of the ASAE 28*, 3 (1985), 755–762.

[41] Kramer, S., and Marder, M. Evolution of river networks. *Physical Review Letters 68* (Jan 1992), 205–208.

[42] Mandelbrot, B. B., and Van Ness, J. W. Fractional brownian motions, fractional noises and applications. *SIAM review 10*, 4 (1968), 422–437.

[43] Mandelbrot, B. B., and Wallis, J. R. Noah, joseph, and operational hydrology. *Water Resources Research 4*, 5 (1968), 909–918.

[44] Mandelbrot, B. B., and Wallis, J. R. Computer experiments with fractional gaussian noises: Part 1, averages and variances. *Water Resources Research 5*, 1 (1969), 228–241.

[45] Massopust, P. R. Vector-valued fractal interpolation functions and their box dimension. *aequationes mathematicae 42*, 1 (1991), 1–22.

[46] Massopust, P. R. *Fractal functions, fractal surfaces, and wavelets.* Academic Press, 2016.

[47] Maysz, R. Convergence of trajectories in fractal interpolation of stochastic processes. *Chaos, Solitons & Fractals 27*, 5, 1328 – 1338.

[48] Moin, P. *Fundamentals of engineering numerical analysis.* Cambridge University Press, New York, N.Y., 2010.

[49] Musgrave, F. K., Kolb, C. E., and Mace, R. S. The synthesis and rendering of eroded fractal terrains. *SIGGRAPH Comput. Graph. 23*, 3 (jul 1989), 41–50.

[50] Parker, G., and Izumi, N. Purely erosional cyclic and solitary steps created by flow over a cohesive bed. *Journal of Fluid Mechanics 419* (2000), 203–238.

[51] Penn, A. I., and Loew, M. H. Estimating fractal dimension with fractal interpolation function models. *IEEE transactions on medical imaging 16*, 6 (1997), 930–937.

[52] Perlin, K. An image synthesizer. *SIGGRAPH Comput. Graph. 19*, 3 (jul 1985), 287–296.

[53] Perron, J. T., Dietrich, W. E., and Kirchner, J. W. Controls on the spacing of first-order valleys. *Journal of Geophysical Research: Earth Surface 113*, F4 (2008).

[54] Perron, J. T., Kirchner, J. W., and Dietrich, W. E. Spectral signatures of characteristic spatial scales and nonfractal structure in landscapes. *Journal of Geophysical Research: Earth Surface 113*, F4 (2008).

[55] Roache, P. J. *Computational Fluid Dynamics*. Hermosa Publishers, Albuquerque, N.M., 1982.

[56] Simpson, G., and Castelltort, S. Coupled model of surface water flow, sediment transport and morphological evolution. *Computers & Geosciences 32*, 10 (2006), 1600–1614.

[57] Simpson, G., and Schlunegger, F. Topographic evolution and morphology of surfaces evolving in response to coupled fluvial and hillslope sediment transport. *Journal of Geophysical Research: Solid Earth 108*, B6 (2003).

[58] Smith, T. R. A theory for the emergence of channelized drainage. *Journal of Geophysical Research: Earth Surface (2003–2012) 115*, F2 (2010).

[59] Smith, T. R., Birnir, B., and Merchant, G. E. Towards an elementary theory of drainage basin evolution: I. the theoretical basis. *Computers & Geosciences 23*, 8 (1997), 811 – 822.

[60] Smith, T. R., and Bretherton, F. P. Stability and the conservation of mass in drainage basin evolution. *Water Resources Research 8*, 6 (1972), 1506–1529.

[61] Smith, T. R., and Merchant, G. E. Conservation principles and the initiation of channelized surface flows. *Natural and Anthropogenic Influences in Fluvial Geomorphology* (1995), 1–25.

[62] Smith, T. R., Merchant, G. E., and Birnir, B. Towards an elementary theory of drainage basin evolution: II. a computational evaluation. *Computers & Geosciences 23*, 8 (1997), 823 – 849.

[63] Todini, E., and Venutelli, M. *Overland flow: a two-dimensional modeling approach*, vol. 345. Springer, 1991, pp. 153–166.

[64] Voss, R. F. Random fractal forgeries. In *Fundamental algorithms for computer graphics*. Springer, 1985, pp. 805–835.

[65] Voss, R. F. Random fractals: Self-affinity in noise, music, mountains, and clouds. *Physica D: Nonlinear Phenomena 38*, 1 (1989), 362–371.

[66] Weissel, J. K., Pratson, L. F., and Malinverno, A. The length-scaling properties of topography. *Journal of Geophysical Research 99* (1994), 13–997.

[67] Welsh, E., Birnir, B., and Bertozzi, A. Shocks in the evolution of an eroding channel. *Applied Mathematics Research eXpress 2006* (2006), 71638.

[68] Wen, R., and Sinding-Larsen, R. Uncertainty in fractal dimension estimated from power spectra and variograms. *Mathematical Geology 29*, 6 (1997), 727–753.

[69] Zhang, W., and Cundy, T. W. Modeling of two-dimensional overland flow. *Water Resources Research 25*, 9 (1989), 2019–2035.

Appendix A

Factorization of the Sediment Flow Equation

The intent of this appendix is to describe the discretization of the sediment flow equation,

$$\frac{\partial H}{\partial t} = \nabla \cdot \left[\frac{\nabla H}{|\nabla H|} h^{\frac{5}{3}\gamma} |\nabla H|^{\frac{\gamma}{2}+\delta} \right] \tag{A.1}$$

into a finite difference scheme. This equation will be discretized by using the Crank-Nicolson scheme and the factoring method to create a series of one-dimensional, tridiagonal systems.

To simplify the discretization process, let $B = \frac{5}{3}\gamma$ and $C = \frac{\gamma}{2} + \delta - 1$. Then Eq. (A.1) becomes

$$\frac{\partial H}{\partial t} = \frac{\partial}{\partial x}\left[h^B H_x (H_x^2 + H_y^2)^{C/2} \right] + \frac{\partial}{\partial y}\left[h^B H_y (H_x^2 + H_y^2)^{C/2} \right] \tag{A.2}$$

where H_x and H_y are the partial derivatives of H in x and y, respectively. Let δ_x and δ_y represent the desired finite difference approximations for the derivatives in x and y,

respectively. Specifically, let

$$H^n_{i\,j,x} \approx \delta_x(H^n_{i\,j}) = \frac{H^n_{i+\frac{1}{2}\,j} - H^n_{i-\frac{1}{2}\,j}}{\Delta x} + O(\Delta x^2)$$

and

$$H^n_{i\,j,y} \approx \delta_y(H^n_{i\,j}) = \frac{H^n_{i\,j+\frac{1}{2}} - H^n_{i\,j-\frac{1}{2}}}{\Delta y} + O(\Delta y^2)$$

be the finite difference schemes. Then using the Crank-Nicolson method, Eq. (A.2) expands to

$$\begin{aligned}\frac{H^{n+1}_{i\,j} - H^n_{i\,j}}{\Delta t} = {} & \frac{E}{2}\delta_x\left((h_{i\,j})^B H^{n+1}_{i\,j,x}\left[\left(H^{n+1}_{i\,j,x}\right)^2 + \left(H^{n+1}_{i\,j,y}\right)^2\right]^{C/2}\right) \\ & + \frac{E}{2}\delta_x\left((h_{i\,j})^B H^{n}_{i\,j,x}\left[\left(H^{n}_{i\,j,x}\right)^2 + \left(H^{n}_{i\,j,y}\right)^2\right]^{C/2}\right) \\ & + \frac{E}{2}\delta_y\left((h_{i\,j})^B H^{n+1}_{i\,j,y}\left[\left(H^{n+1}_{i\,j,x}\right)^2 + \left(H^{n+1}_{i\,j,y}\right)^2\right]^{C/2}\right) \\ & + \frac{E}{2}\delta_y\left((h_{i\,j})^B H^{n}_{i\,j,y}\left[\left(H^{n}_{i\,j,x}\right)^2 + \left(H^{n}_{i\,j,y}\right)^2\right]^{C/2}\right) + O(\Delta x^2, \Delta t^2). \quad \text{(A.3)}\end{aligned}$$

There is no time step associated to the water depth h as it is assumed to be in a statistically stationary state when Eq. (A.1) is solved, i.e., $h^n = h^{n+1} = h$. The constant E represents an erosion coefficient introduced by the Manning type relation in the initial sediment flow relation. In Eq. (A.3) there are also two nonlinear terms in H^{n+1}, namely

$$H^{n+1}_{i\,j,x}\left[\left(H^{n+1}_{i\,j,x}\right)^2 + \left(H^{n+1}_{i\,j,y}\right)^2\right]^{C/2}$$

and

$$H^{n+1}_{i\,j,y}\left[\left(H^{n+1}_{i\,j,x}\right)^2 + \left(H^{n+1}_{i\,j,y}\right)^2\right]^{C/2}.$$

To rectify this situation, let

$$f(t_n) = H^n_{ij,x}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2}$$

and let $t_{n+1} = t_n + \Delta t$. Then by Taylor's Theorem,

$$\begin{aligned} f(t_{n+1}) &\approx f(t_n) + \Delta t \frac{\partial f(t_n)}{\partial t} \\ &= H^n_{ij,x}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2} + \Delta t\Bigg\{\frac{\partial H^n_{ij,x}}{\partial t}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2} \\ &\qquad + CH^n_{ij,x}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2-1}\left[H^n_{ij,x}\frac{\partial H^n_{ij,x}}{\partial t} + H^n_{ij,y}\frac{\partial H^n_{ij,y}}{\partial t}\right]\Bigg\}. \end{aligned}$$

The time derivatives of the spacial derivatives in H can be approximated by

$$\frac{\partial H^n_{ij,x}}{\partial t} = \frac{H^{n+1}_{ij,x} - H^n_{ij,x}}{\Delta t} \quad \text{and} \quad \frac{\partial H^n_{ij,y}}{\partial t} = \frac{H^{n+1}_{ij,y} - H^n_{ij,y}}{\Delta t}.$$

Thus,

$$\begin{aligned} H^{n+1}_{ij,x}\left[\left(H^{n+1}_{ij,x}\right)^2 + \left(H^{n+1}_{ij,y}\right)^2\right]^{C/2} &\approx H^{n+1}_{ij,x}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2} \\ &\quad + CH^n_{ij,x}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2-1}\left[H^n_{ij,x}\left(H^{n+1}_{ij,x} - H^n_{ij,x}\right)\right] \\ &\quad + CH^n_{ij,x}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2-1}\left[H^n_{ij,y}\left(H^{n+1}_{ij,y} - H^n_{ij,y}\right)\right] \end{aligned}$$

and similarly,

$$\begin{aligned} H^{n+1}_{ij,y}\left[\left(H^{n+1}_{ij,x}\right)^2 + \left(H^{n+1}_{ij,y}\right)^2\right]^{C/2} &\approx H^{n+1}_{ij,y}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2} \\ &\quad + CH^n_{ij,y}\left[\left(H^n_{ij,x}\right)^2 + \left(H^n_{ij,y}\right)^2\right]^{C/2-1}\left[H^n_{ij,x}\left(H^{n+1}_{ij,x} - H^n_{ij,x}\right)\right] \end{aligned}$$

$$+ CH^n_{i\,j,y}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j,y}\left(H^{n+1}_{i\,j,y} - H^n_{i\,j,y}\right)\right]$$

are the desired linearizations. Substituting these into Eq. (A.3) yields the linear discretization

$$\begin{aligned}
\frac{H^{n+1}_{ij} - H^n_{ij}}{\Delta t} = \frac{E}{2}\delta_x\Bigg((h_{i\,j})^B\Bigg(&H^{n+1}_{i\,j,x}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2} \\
&+ CH^n_{i\,j,x}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j,x}\left(H^{n+1}_{i\,j,x} - H^n_{i\,j,x}\right)\right] \\
&+ CH^n_{i\,j,x}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j,y}\left(H^{n+1}_{i\,j,y} - H^n_{i\,j,y}\right)\right]\Bigg)\Bigg) \\
+ \frac{E}{2}\delta_y\Bigg((h_{i\,j})^B\Bigg(&H^{n+1}_{i\,j,y}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2} \\
&+ CH^n_{i\,j,y}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j,x}\left(H^{n+1}_{i\,j,x} - H^n_{i\,j,x}\right)\right] \\
&+ CH^n_{i\,j,y}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j,y}\left(H^{n+1}_{i\,j,y} - H^n_{i\,j,y}\right)\right]\Bigg)\Bigg) \\
+ \frac{E}{2}\delta_x\left((h_{i\,j})^B H^n_{i\,j,x}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2}\right) \\
+ \frac{E}{2}\delta_y\left((h_{i\,j})^B H^n_{i\,j,y}\left[\left(H^n_{i\,j,x}\right)^2 + \left(H^n_{i\,j,y}\right)^2\right]^{C/2}\right) + O(\Delta x^2, \Delta t^2). \quad \text{(A.4)}
\end{aligned}$$

Now apply the definitions of the δ_x and δ_y operators.

$$\begin{aligned}
\frac{H^{n+1}_{ij} - H^n_{ij}}{\Delta t} = \frac{E}{2\Delta x}\Bigg\{(h_{i+\frac{1}{2}\,j})^B\Bigg(&H^{n+1}_{i+\frac{1}{2}\,j,x}\left[\left(H^n_{i+\frac{1}{2}\,j,x}\right)^2 + \left(H^n_{i+\frac{1}{2}\,j,y}\right)^2\right]^{C/2} \\
&+ CH^n_{i+\frac{1}{2}\,j,x}\left[\left(H^n_{i+\frac{1}{2}\,j,x}\right)^2 + \left(H^n_{i+\frac{1}{2}\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i+\frac{1}{2}\,j,x}\left(H^{n+1}_{i+\frac{1}{2}\,j,x} - H^n_{i+\frac{1}{2}\,j,x}\right)\right] \\
&+ CH^n_{i+\frac{1}{2}\,j,x}\left[\left(H^n_{i+\frac{1}{2}\,j,x}\right)^2 + \left(H^n_{i+\frac{1}{2}\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i+\frac{1}{2}\,j,y}\left(H^{n+1}_{i+\frac{1}{2}\,j,y} - H^n_{i+\frac{1}{2}\,j,y}\right)\right]\Bigg) \\
&- (h_{i-\frac{1}{2}\,j})^B\Bigg(H^{n+1}_{i-\frac{1}{2}\,j,x}\left[\left(H^n_{i-\frac{1}{2}\,j,x}\right)^2 + \left(H^n_{i-\frac{1}{2}\,j,y}\right)^2\right]^{C/2}
\end{aligned}$$

$$
\begin{aligned}
&+ CH^n_{i-\frac{1}{2}\,j,x}\left[\left(H^n_{i-\frac{1}{2}\,j,x}\right)^2+\left(H^n_{i-\frac{1}{2}\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i-\frac{1}{2}\,j,x}\left(H^{n+1}_{i-\frac{1}{2}\,j,x}-H^n_{i-\frac{1}{2}\,j,x}\right)\right]\\
&+ CH^n_{i-\frac{1}{2}\,j,x}\left[\left(H^n_{i-\frac{1}{2}\,j,x}\right)^2+\left(H^n_{i-\frac{1}{2}\,j,y}\right)^2\right]^{C/2-1}\left[H^n_{i-\frac{1}{2}\,j,y}\left(H^{n+1}_{i-\frac{1}{2}\,j,y}-H^n_{i-\frac{1}{2}\,j,y}\right)\right]\Bigg)\Bigg\}\\
&+\frac{E}{2\Delta y}\Bigg\{(h_{i\,j+\frac{1}{2}})^B\Bigg(H^{n+1}_{i\,j+\frac{1}{2},x}\left[\left(H^n_{i\,j+\frac{1}{2},x}\right)^2+\left(H^n_{i\,j+\frac{1}{2},y}\right)^2\right]^{C/2}\\
&+ CH^n_{i\,j+\frac{1}{2},x}\left[\left(H^n_{i\,j+\frac{1}{2},x}\right)^2+\left(H^n_{i\,j+\frac{1}{2},y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j+\frac{1}{2},x}\left(H^{n+1}_{i\,j+\frac{1}{2},x}-H^n_{i\,j+\frac{1}{2},x}\right)\right]\\
&+ CH^n_{i\,j+\frac{1}{2},x}\left[\left(H^n_{i\,j+\frac{1}{2},x}\right)^2+\left(H^n_{i\,j+\frac{1}{2},y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j+\frac{1}{2},y}\left(H^{n+1}_{i\,j+\frac{1}{2},y}-H^n_{i\,j+\frac{1}{2},y}\right)\right]\Bigg)\\
&-(h_{i\,j-\frac{1}{2}})^B\Bigg(H^{n+1}_{i\,j-\frac{1}{2},x}\left[\left(H^n_{i\,j-\frac{1}{2},x}\right)^2+\left(H^n_{i\,j-\frac{1}{2},y}\right)^2\right]^{C/2}\\
&+ CH^n_{i\,j-\frac{1}{2},x}\left[\left(H^n_{i\,j-\frac{1}{2},x}\right)^2+\left(H^n_{i\,j-\frac{1}{2},y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j-\frac{1}{2},x}\left(H^{n+1}_{i\,j-\frac{1}{2},x}-H^n_{i\,j-\frac{1}{2},x}\right)\right]\\
&+ CH^n_{i\,j-\frac{1}{2},x}\left[\left(H^n_{i\,j-\frac{1}{2},x}\right)^2+\left(H^n_{i\,j-\frac{1}{2},y}\right)^2\right]^{C/2-1}\left[H^n_{i\,j-\frac{1}{2},y}\left(H^{n+1}_{i\,j-\frac{1}{2},y}-H^n_{i\,j-\frac{1}{2},y}\right)\right]\Bigg)\Bigg\}\\
&+\frac{E}{2\Delta x}\Bigg\{(h_{i+\frac{1}{2}\,j})^B H^n_{i+\frac{1}{2}\,j,x}\left[\left(H^n_{i+\frac{1}{2}\,j,x}\right)^2+\left(H^n_{i+\frac{1}{2}\,j,y}\right)^2\right]^{C/2}\\
&-(h_{i-\frac{1}{2}\,j})^B H^n_{i-\frac{1}{2}\,j,x}\left[\left(H^n_{i-\frac{1}{2}\,j,x}\right)^2+\left(H^n_{i-\frac{1}{2}\,j,y}\right)^2\right]^{C/2}\Bigg\}\\
&+\frac{E}{2\Delta y}\Bigg\{(h_{i\,j+\frac{1}{2}})^B H^n_{i\,j+\frac{1}{2},y}\left[\left(H^n_{i\,j+\frac{1}{2},x}\right)^2+\left(H^n_{i\,j+\frac{1}{2},y}\right)^2\right]^{C/2}\\
&-(h_{i\,j-\frac{1}{2}})^B H^n_{i\,j-\frac{1}{2},y}\left[\left(H^n_{i\,j-\frac{1}{2},x}\right)^2+\left(H^n_{i\,j-\frac{1}{2},y}\right)^2\right]^{C/2}\Bigg\}+O(\Delta x^2,\Delta t^2). \quad \text{(A.5)}
\end{aligned}
$$

This form is long, complicated, and contains multiple repeated calculations. Before expanding Eq. (A.5) any further, the following table of variables/simplifications will be implemented:

Table 1

$$X_0 = \frac{H^n_{i+1\,j} - H^n_{i-1\,j}}{2\Delta x}$$

$$X_1 = \frac{H^n_{i+1\,j+1} + H^n_{i\,j+1} - H^n_{i\,j-1} - H^n_{i+1\,j-1}}{4\Delta x} \qquad X_2 = \frac{H^n_{i-1\,j+1} + H^n_{i\,j+1} - H^n_{i\,j-1} - H^n_{i-1\,j-1}}{4\Delta x}$$

$$X_3 = \left(\frac{h^n_{i+1\,j} + h^n_{i\,j}}{2}\right)^B \quad X_4 = \left(\frac{h^n_{i\,j} + h^n_{i-1\,j}}{2}\right)^B \quad X_5 = \frac{H^n_{i+1\,j} - H^n_{i\,j}}{\Delta x} \quad X_6 = \frac{H^n_{i\,j} - H^n_{i-1\,j}}{\Delta x}$$

$$X_7 = X_1^2 + X_5^2 \qquad X_8 = X_7^{C/2-1} \qquad X_9 = X_7 X_8 \qquad X_{10} = X_2^2 + X_6^2$$

$$X_{11} = X_{10}^{C/2-1} \qquad X_{12} = X_{10} X_{11} \qquad X_{13} = X_3 X_9 \qquad X_{14} = X_4 X_{12}$$

$$X_{15} = C X_3 X_5 X_8 \qquad X_{16} = X_5 X_{15} \qquad X_{17} = X_1 X_{15} \qquad X_{18} = C X_4 X_6 X_{11}$$

$$X_{19} = X_6 X_{18} \qquad X_{20} = X_2 X_{18}$$

$$Y_0 = \frac{H^n_{i\,j+1} - H^n_{i\,j-1}}{2\Delta x}$$

$$Y_1 = \frac{H^n_{i+1\,j+1} + H^n_{i+1\,j} - H^n_{i-1\,j} - H^n_{i-1\,j+1}}{4\Delta x} \qquad Y_2 = \frac{H^n_{i+1\,j-1} + H^n_{i+1\,j} - H^n_{i-1\,j} - H^n_{i-1\,j-1}}{4\Delta x}$$

$$Y_3 = \left(\frac{h^n_{i\,j+1} + h^n_{i\,j}}{2}\right)^B \quad Y_4 = \left(\frac{h^n_{i\,j} + h^n_{i\,j-1}}{2}\right)^B \quad Y_5 = \frac{H^n_{i\,j+1} - H^n_{i\,j}}{\Delta x} \quad Y_6 = \frac{H^n_{i\,j} - H^n_{i\,j-1}}{\Delta x}$$

$$Y_7 = Y_1^2 + Y_5^2 \qquad Y_8 = Y_7^{C/2-1} \qquad Y_9 = Y_7 Y_8 \qquad Y_{10} = Y_2^2 + Y_6^2$$

$$Y_{11} = Y_{10}^{C/2-1} \qquad Y_{12} = Y_{10} Y_{11} \qquad Y_{13} = Y_3 Y_9 \qquad Y_{14} = Y_4 Y_{12}$$

$$Y_{15} = C Y_3 Y_5 Y_8 \qquad Y_{16} = Y_5 Y_{15} \qquad Y_{17} = Y_1 Y_{15} \qquad Y_{18} = C Y_4 Y_6 Y_{11}$$

$$Y_{19} = Y_6 Y_{18} \qquad Y_{20} = Y_2 Y_{18}$$

Notice that the implementation of this table makes a very crucial assumption: that the grid spacing is uniform (i.e., $\Delta x = \Delta y$).

Applying this table of simplifications results in

$$\begin{aligned}\frac{H_{ij}^{n+1}-H_{ij}^{n}}{\Delta t} = \frac{E}{2\Delta x}\Bigg\{ & X_{16}\left[H_{i+\frac{1}{2}\,j,x}^{n+1}-H_{i+\frac{1}{2}\,j,x}^{n}\right]+X_{17}\left[H_{i+\frac{1}{2}\,j,y}^{n+1}-H_{i+\frac{1}{2}\,j,y}^{n}\right]\\ & -X_{19}\left[H_{i-\frac{1}{2}\,j,x}^{n+1}-H_{i-\frac{1}{2}\,j,x}^{n}\right]-X_{20}\left[H_{i-\frac{1}{2}\,j,y}^{n+1}-H_{i-\frac{1}{2}\,j,y}^{n}\right]\\ & +Y_{17}\left[H_{i\,j+\frac{1}{2},x}^{n+1}-H_{i\,j+\frac{1}{2},x}^{n}\right]+Y_{16}\left[H_{i\,j+\frac{1}{2},y}^{n+1}-H_{i\,j+\frac{1}{2},y}^{n}\right]\\ & -Y_{20}\left[H_{i\,j-\frac{1}{2},x}^{n+1}-H_{i\,j-\frac{1}{2},x}^{n}\right]-Y_{19}\left[H_{i\,j-\frac{1}{2},y}^{n+1}-H_{i\,j-\frac{1}{2},y}^{n}\right]\Bigg\}\\ & +\frac{E}{2\Delta x}\Bigg\{X_{13}\left[H_{i+\frac{1}{2}\,j,x}^{n+1}+H_{i+\frac{1}{2}\,j,x}^{n}\right]-X_{14}\left[H_{i-\frac{1}{2}\,j,x}^{n+1}+H_{i-\frac{1}{2}\,j,x}^{n}\right]\\ & +Y_{13}\left[H_{i\,j+\frac{1}{2},y}^{n+1}+H_{i\,j+\frac{1}{2},y}^{n}\right]-Y_{14}\left[H_{i\,j-\frac{1}{2},y}^{n+1}+H_{i\,j-\frac{1}{2},y}^{n}\right]\Bigg\}. \quad \text{(A.6)}\end{aligned}$$

Now expand all of the derivative operators,

$$\begin{aligned}&\frac{H_{ij}^{n+1}-H_{ij}^{n}}{\Delta t} =\\ &\frac{E}{2\Delta x}\Bigg\{X_{16}\left[\frac{H_{i+1\,j}^{n+1}-H_{ij}^{n+1}}{\Delta x}-\frac{H_{i+1\,j}^{n}-H_{ij}^{n}}{\Delta x}\right]-X_{19}\left[\frac{H_{ij}^{n+1}-H_{i-1\,j}^{n+1}}{\Delta x}-\frac{H_{ij}^{n}-H_{i-1\,j}^{n}}{\Delta x}\right]\\ &+X_{17}\left[\frac{H_{i+1\,j+1}^{n+1}+H_{ij+1}^{n+1}-H_{ij-1}^{n+1}-H_{i+1\,j-1}^{n+1}}{4\Delta x}-\frac{H_{i+1\,j+1}^{n}+H_{ij+1}^{n}-H_{ij-1}^{n}-H_{i+1\,j-1}^{n}}{4\Delta x}\right]\\ &-X_{20}\left[\frac{H_{i-1\,j+1}^{n+1}+H_{ij+1}^{n+1}-H_{ij-1}^{n+1}-H_{i-1\,j-1}^{n+1}}{\Delta x}-\frac{H_{i-1\,j+1}^{n}+H_{ij+1}^{n}-H_{ij-1}^{n}-H_{i-1\,j-1}^{n}}{\Delta x}\right]\\ &\quad+Y_{16}\left[\frac{H_{ij+1}^{n+1}-H_{ij}^{n+1}}{\Delta x}-\frac{H_{ij+1}^{n}-H_{ij}^{n}}{\Delta x}\right]-Y_{19}\left[\frac{H_{ij}^{n+1}-H_{ij-1}^{n+1}}{\Delta x}-\frac{H_{ij}^{n}-H_{ij-1}^{n}}{\Delta x}\right]\\ &+Y_{17}\left[\frac{H_{i+1\,j+1}^{n+1}+H_{i+1\,j}^{n+1}-H_{i-1\,j}^{n+1}-H_{i-1\,j+1}^{n+1}}{4\Delta x}-\frac{H_{i+1\,j+1}^{n}+H_{i+1\,j}^{n}-H_{i-1\,j}^{n}-H_{i-1\,j+1}^{n}}{4\Delta x}\right]\\ &-Y_{20}\left[\frac{H_{i+1\,j-1}^{n+1}+H_{i+1\,j}^{n+1}-H_{i-1\,j}^{n+1}-H_{i-1\,j-1}^{n+1}}{4\Delta x}-\frac{H_{i+1\,j-1}^{n}+H_{i+1\,j}^{n}-H_{i-1\,j}^{n}-H_{i-1\,j-1}^{n}}{4\Delta x}\right]\end{aligned}$$

$$+ X_{13}\left[\frac{H^{n+1}_{i+1\,j} - H^{n+1}_{ij}}{\Delta x} + \frac{H^{n}_{i+1\,j} - H^{n}_{ij}}{\Delta x}\right] - X_{14}\left[\frac{H^{n+1}_{ij} - H^{n+1}_{i-1\,j}}{\Delta x} + \frac{H^{n}_{ij} - H^{n}_{i-1\,j}}{\Delta x}\right]$$
$$+ Y_{13}\left[\frac{H^{n+1}_{ij+1} - H^{n+1}_{ij}}{\Delta x} + \frac{H^{n}_{ij+1} - H^{n}_{ij}}{\Delta x}\right] - Y_{14}\left[\frac{H^{n+1}_{ij} - H^{n+1}_{ij-1}}{\Delta x} + \frac{H^{n}_{ij} - H^{n}_{ij-1}}{\Delta x}\right]\Bigg\} \quad \text{(A.7)}$$

Combine terms of H that are at the same grid cells.

$$\begin{aligned}\frac{H^{n+1}_{ij} - H^{n}_{ij}}{\Delta t} = \frac{E}{2(\Delta x)^2}\Bigg\{ & X_{13}\left[H^{n+1}_{i+1\,j} + H^{n}_{i+1\,j}\right] + \left(X_{16} + \frac{1}{4}Y_{17} - \frac{1}{4}Y_{20}\right)\left[H^{n+1}_{i+1\,j} - H^{n}_{i+1\,j}\right] \\ & + X_{14}\left[H^{n+1}_{i-1\,j} + H^{n}_{i-1\,j}\right] + \left(X_{19} - \frac{1}{4}Y_{17} + \frac{1}{4}Y_{20}\right)\left[H^{n+1}_{i-1\,j} - H^{n}_{i-1\,j}\right] \\ & + Y_{13}\left[H^{n+1}_{ij+1} + H^{n}_{ij+1}\right] + \left(Y_{16} + \frac{1}{4}X_{17} - \frac{1}{4}X_{20}\right)\left[H^{n+1}_{ij+1} - H^{n}_{ij+1}\right] \\ & + Y_{14}\left[H^{n+1}_{ij-1} + H^{n}_{ij-1}\right] + \left(Y_{19} - \frac{1}{4}X_{17} + \frac{1}{4}X_{20}\right)\left[H^{n+1}_{ij-1} - H^{n}_{ij-1}\right] \\ & + \left(\frac{1}{4}X_{17} + \frac{1}{4}Y_{17}\right)\left[H^{n+1}_{i+1\,j+1} - H^{n}_{i+1\,j+1}\right] + \left(\frac{1}{4}X_{20} + \frac{1}{4}Y_{20}\right)\left[H^{n+1}_{i-1\,j-1} - H^{n}_{i-1\,j-1}\right] \\ & - \left(\frac{1}{4}X_{17} + \frac{1}{4}Y_{20}\right)\left[H^{n+1}_{i+1\,j-1} - H^{n}_{i+1\,j-1}\right] - \left(\frac{1}{4}X_{20} + \frac{1}{4}Y_{17}\right)\left[H^{n+1}_{i-1\,j+1} - H^{n}_{i-1\,j+1}\right] \\ & - \left(X_{13} + X_{14} + Y_{13} + Y_{14}\right)\left[H^{n+1}_{ij} + H^{n}_{ij}\right] - \left(X_{16} + X_{19} + Y_{16} + Y_{19}\right)\left[H^{n+1}_{ij} - H^{n}_{ij}\right]\Bigg\}\end{aligned} \quad \text{(A.8)}$$

Next add zeros to this equation so that all terms are of the form $H^{n+1} - H^{n}$.

$$\begin{aligned}\frac{H^{n+1}_{ij} - H^{n}_{ij}}{\Delta t} = \frac{E}{2(\Delta x)^2}\Bigg\{ & \left(X_{13} + X_{16} + \frac{1}{4}Y_{17} - \frac{1}{4}Y_{20}\right)\left[H^{n+1}_{i+1\,j} - H^{n}_{i+1\,j}\right] \\ & + \left(X_{14} + X_{19} - \frac{1}{4}Y_{17} + \frac{1}{4}Y_{20}\right)\left[H^{n+1}_{i-1\,j} - H^{n}_{i-1\,j}\right] \\ & + \left(Y_{13} + Y_{16} + \frac{1}{4}X_{17} - \frac{1}{4}X_{20}\right)\left[H^{n+1}_{ij+1} - H^{n}_{ij+1}\right] \\ & + \left(Y_{14} + Y_{19} - \frac{1}{4}X_{17} + \frac{1}{4}X_{20}\right)\left[H^{n+1}_{ij-1} - H^{n}_{ij-1}\right]\end{aligned}$$

$$
\begin{aligned}
&+\left(\frac{1}{4}X_{17}+\frac{1}{4}Y_{17}\right)\left[H^{n+1}_{i+1\,j+1}-H^{n}_{i+1\,j+1}\right]+\left(\frac{1}{4}X_{20}+\frac{1}{4}Y_{20}\right)\left[H^{n+1}_{i-1\,j-1}-H^{n}_{i-1\,j-1}\right]\\
&-\left(\frac{1}{4}X_{17}+\frac{1}{4}Y_{20}\right)\left[H^{n+1}_{i+1\,j-1}-H^{n}_{i+1\,j-1}\right]-\left(\frac{1}{4}X_{20}+\frac{1}{4}Y_{17}\right)\left[H^{n+1}_{i-1\,j+1}-H^{n}_{i-1\,j+1}\right]\\
&\qquad-\left(X_{13}+X_{14}+X_{16}+X_{19}+Y_{13}+Y_{14}+Y_{16}+Y_{19}\right)\left[H^{n+1}_{ij}-H^{n}_{ij}\right]\Bigg\}\\
&\qquad+\frac{E}{(\Delta x)^2}\Bigg\{X_{13}H^{n}_{i+1\,j}+X_{14}H^{n}_{i-1\,j}+Y_{13}H^{n}_{i\,j+1}+Y_{14}H^{n}_{i\,j-1}\\
&\qquad\qquad-\left(X_{13}+X_{14}+Y_{13}+Y_{14}\right)H^{n}_{ij}\Bigg\} \qquad \text{(A.9)}
\end{aligned}
$$

Now we re-arrange the terms one last time to from linear operators in the four directions: x, y, and the diagonals $y=\pm x$.

$$
\begin{aligned}
\frac{H^{n+1}_{ij}-H^{n}_{ij}}{\Delta t}=\frac{E}{2\Delta x}\Bigg\{&\left(X_{13}+X_{16}\right)\left[\frac{H^{n+1}_{i+1\,j}-H^{n+1}_{ij}}{\Delta x}-\frac{H^{n}_{i+1\,j}-H^{n}_{ij}}{\Delta x}\right]\\
&+\left(X_{14}+X_{19}\right)\left[\frac{H^{n+1}_{ij}-H^{n+1}_{i-1\,j}}{\Delta x}-\frac{H^{n}_{ij}-H^{n}_{i-1\,j}}{\Delta x}\right]\\
&+\frac{\left(Y_{17}-Y_{20}\right)}{2}\left[\frac{H^{n+1}_{i+1\,j}-H^{n+1}_{i-1\,j}}{2\Delta x}-\frac{H^{n}_{i+1\,j}-H^{n}_{i-1\,j}}{2\Delta x}\right]\\
&+\left(Y_{13}+Y_{16}\right)\left[\frac{H^{n+1}_{i\,j+1}-H^{n+1}_{ij}}{\Delta x}-\frac{H^{n}_{i\,j+1}-H^{n}_{ij}}{\Delta x}\right]\\
&+\left(Y_{14}+Y_{19}\right)\left[\frac{H^{n+1}_{ij}-H^{n+1}_{i\,j-1}}{\Delta x}-\frac{H^{n}_{ij}-H^{n}_{i\,j-1}}{\Delta x}\right]\\
&+\frac{\left(X_{17}-X_{20}\right)}{2}\left[\frac{H^{n+1}_{i\,j+1}-H^{n+1}_{i\,j-1}}{2\Delta x}-\frac{H^{n}_{i\,j+1}-H^{n}_{i\,j-1}}{2\Delta x}\right]\\
&+\frac{X_{17}+Y_{17}}{4\Delta x}\left[H^{n+1}_{i+1\,j+1}-H^{n}_{i+1\,j+1}\right]+\frac{X_{20}+Y_{20}}{4\Delta x}\left[H^{n+1}_{i-1\,j-1}-H^{n}_{i-1\,j-1}\right]\\
&-\frac{X_{17}+Y_{20}}{4\Delta x}\left[H^{n+1}_{i+1\,j-1}-H^{n}_{i+1\,j-1}\right]-\frac{X_{20}+Y_{17}}{4\Delta x}\left[H^{n+1}_{i-1\,j+1}-H^{n}_{i-1\,j+1}\right]\Bigg\}
\end{aligned}
$$

$$+\frac{E}{(\Delta x)^2}\Big\{X_{13}H^n_{i+1\,j}+X_{14}H^n_{i-1\,j}+Y_{13}H^n_{i\,j+1}+Y_{14}H^n_{i\,j-1} - (X_{13}+X_{14}+Y_{13}+Y_{14})\,H^n_{i\,j}\Big\} \quad \text{(A.10)}$$

Define the following linear operators,

$$A_x(H) = (X_{13}+X_{16})\frac{H_{i+1\,j}-H_{i\,j}}{\Delta x} + (X_{14}+X_{19})\frac{H_{i\,j}-H_{i-1\,j}}{\Delta x} + \frac{Y_{17}-Y_{20}}{2}\frac{H_{i+1\,j}-H_{i-1\,j}}{2\Delta x},$$

$$A_y(H) = (Y_{13}+Y_{16})\frac{H_{i\,j+1}-H_{i\,j}}{\Delta x} + (Y_{14}+Y_{19})\frac{H_{i\,j}-H_{i\,j-1}}{\Delta x} + \frac{X_{17}-X_{20}}{2}\frac{H_{i\,j+1}-H_{i\,j-1}}{2\Delta x},$$

$$D_+(H) = \frac{X_{17}+Y_{17}}{4\Delta x}H_{i+1\,j+1} + \frac{X_{20}+Y_{20}}{4\Delta x}H_{i-1\,j-1},$$

$$D_-(H) = -\frac{X_{17}+Y_{20}}{4\Delta x}H_{i+1\,j-1} - \frac{X_{20}+Y_{17}}{4\Delta x}H_{i-1\,j+1},$$

the constant vector

$$v(H) = \frac{1}{\Delta x}\Big\{X_{13}H^n_{i+1\,j}+X_{14}H^n_{i-1\,j}+Y_{13}H^n_{i\,j+1}+Y_{14}H^n_{i\,j-1}-(X_{13}+X_{14}+Y_{13}+Y_{14})\,H^n_{i\,j}\Big\},$$

and define $\lambda = \frac{E\Delta t}{2\Delta x}$.

Using these new definitions, Eq. (A.10) can be written as

$$H^{n+1} = H^n + \lambda\Big\{ A_x(H^{n+1} - H^n) + A_y(H^{n+1} - H^n) + D_+(H^{n+1} - H^n) + D_-(H^{n+1} - H^n) \Big\} + 2\lambda v(H^n) \quad \text{(A.11)}$$

All of the operators are linear, so can be separated by moving all terms in time $n+1$ to the left and side, and leaving all terms at time t_n on the right hand side. This yields

$$H^{n+1} + \lambda A_x(H^{n+1}) + \lambda A_y(H^{n+1}) + \lambda D_+(H^{n+1}) + \lambda D_-(H^{n+1}) = H^n + \lambda A_x(H^n) + \lambda A_y(H^n) + \lambda D_+(H^n) + \lambda D_-(H^n) + 2\lambda v(H^n) \quad \text{(A.12)}$$

Now "factor" both sides of equation (A.12) to get

$$(I + \lambda A_x)(I + \lambda A_y)(I + \lambda D_+)(I + \lambda D_-)H^{n+1} = (I + \lambda A_x)(I + \lambda A_y)(I + \lambda D_+)(I + \lambda D_-)H^n + 2\lambda v(H^n) \quad \text{(A.13)}$$

where I is the identity matrix.

Notice this factorization has introduced new operators not in Eq. (A.12), namely the cross terms, e.g. $\lambda^2 A_x A_y$, $\lambda^2 A_x D_+$, $\lambda^3 A_x A_y D_-$, etc. These cross terms appear on both sides of Eq. (A.13). Multiplying out and simplifying results in terms such as $\lambda^2 A_x A_y(H^{n+1} - H^n)$. By Taylor's Theorem, $H^{n+1} - H^n \approx O(\Delta t)$. Also, $\lambda \approx \frac{\Delta t}{\Delta x}$ in order. Therefore, any cross term has order at least $\frac{(\Delta t)^3}{(\Delta x)^2}$. The value of Δt is significantly smaller than that of Δx. The Crank-Nicoloson scheme is already second order accurate, thus these cross terms of (practical) order $(\Delta t)^3$ or higher have no negative consequences on the accuracy.

Appendix B

Artificial Landscapes

This appendix consists of a series of figures that show the differences in the various methods of adding small scale dynamics, described in Sect. 5.2.5, to a surface. Figure B.1 is a basic artificial surface. The following six figures are the results of adding the different small scale methods to that surface. The last three figures show the results of running three of those surfaces through the numerics.

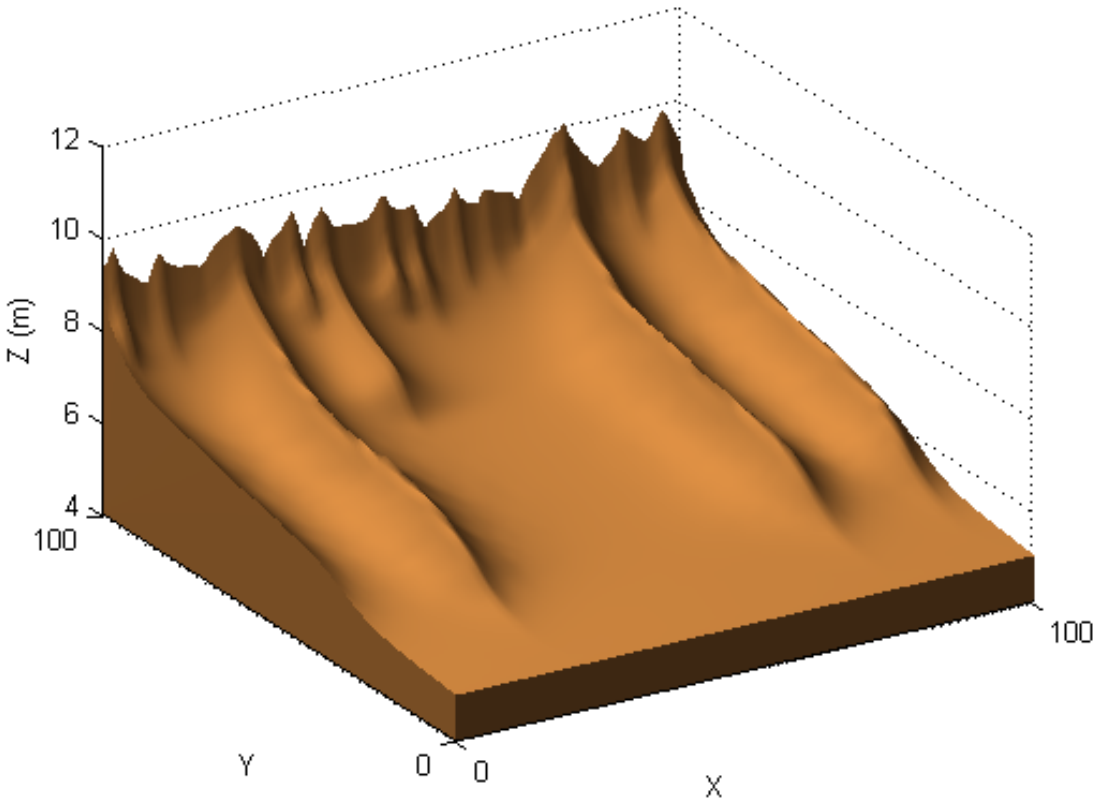


Figure B.1: A basic artificial surface

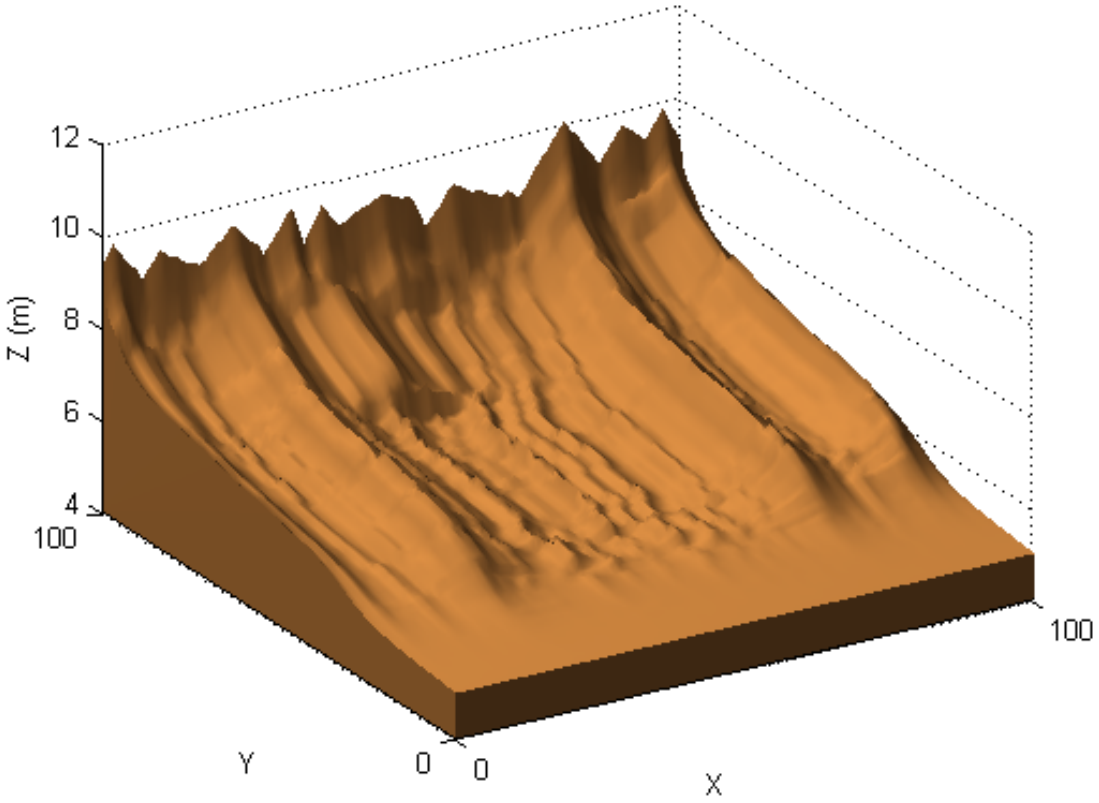


Figure B.2: Small scales added with the lateral (DP) Hurst interpolation

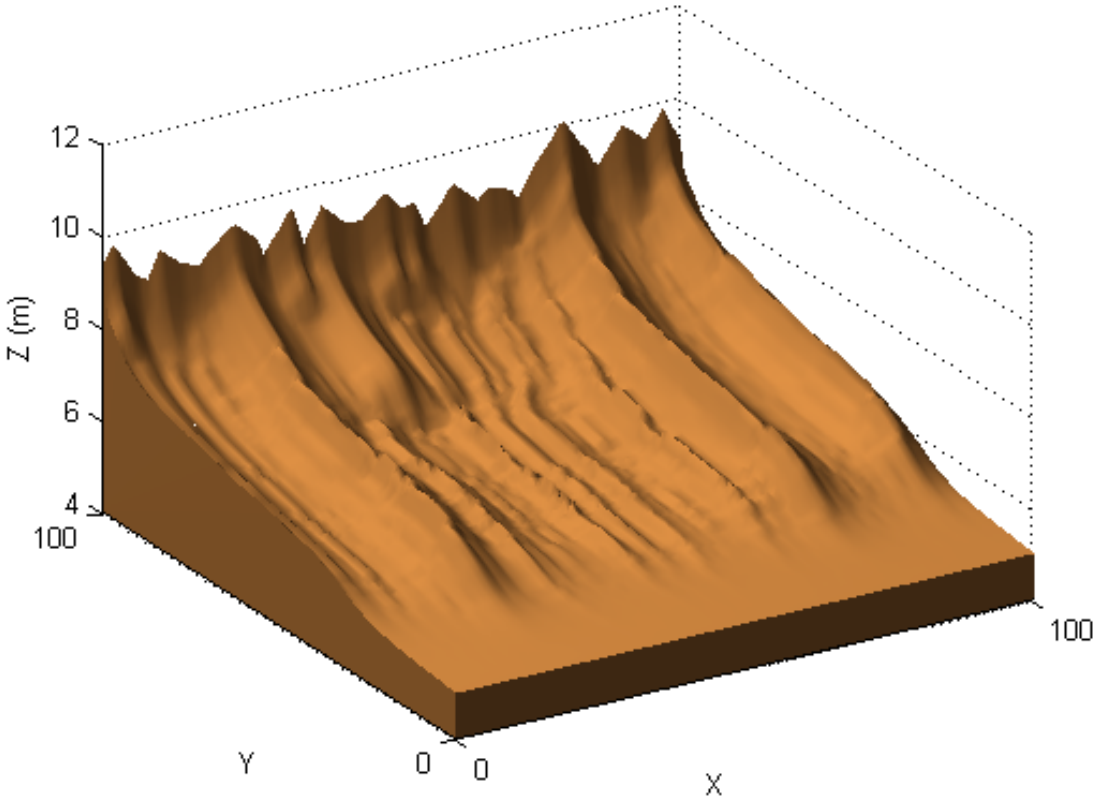


Figure B.3: Small scales added with the lateral (extrema) Hurst interpolation

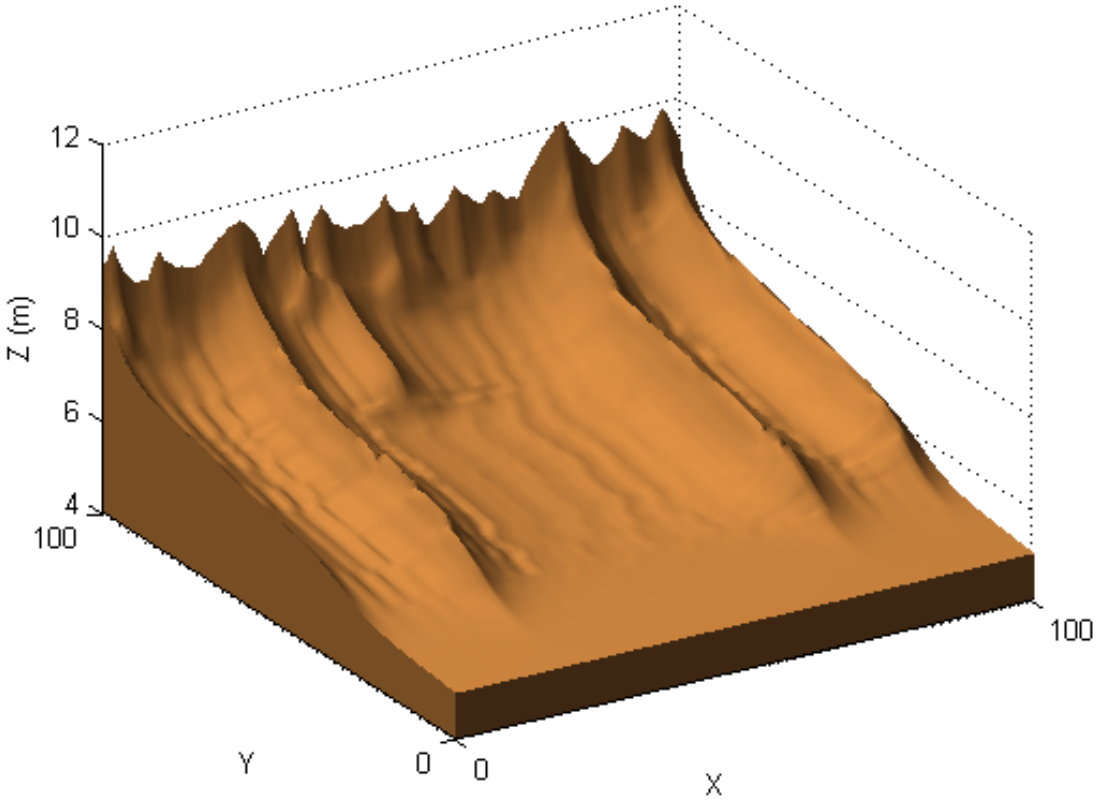


Figure B.4: Small scales added with the piecewise (DP) Hurst interpolation

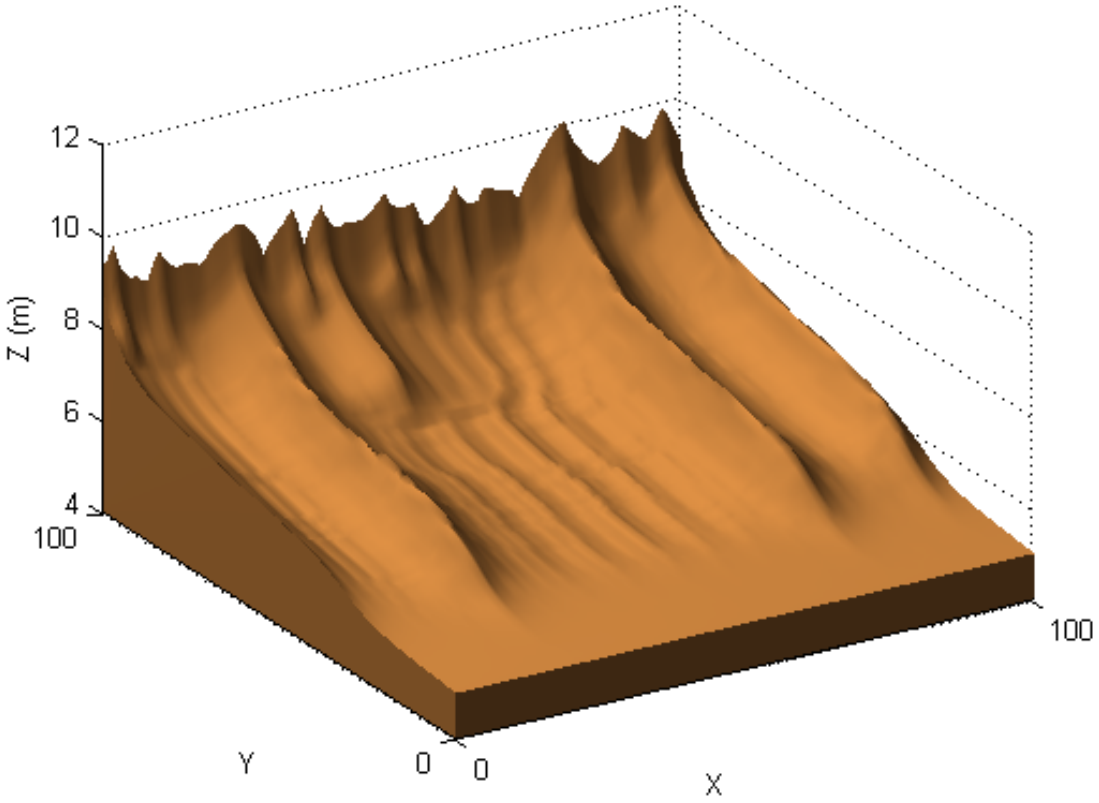


Figure B.5: Small scales added with the piecewise (extrema) Hurst interpolation

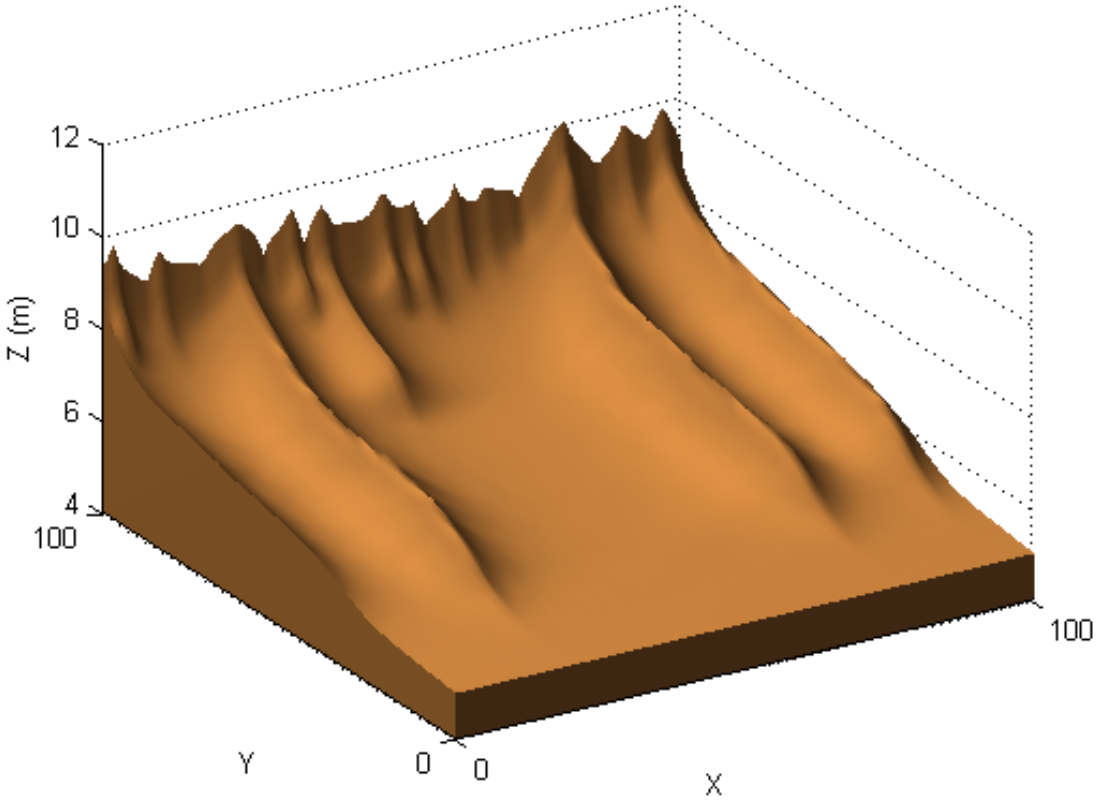


Figure B.6: Small scales added with independent fractal noise

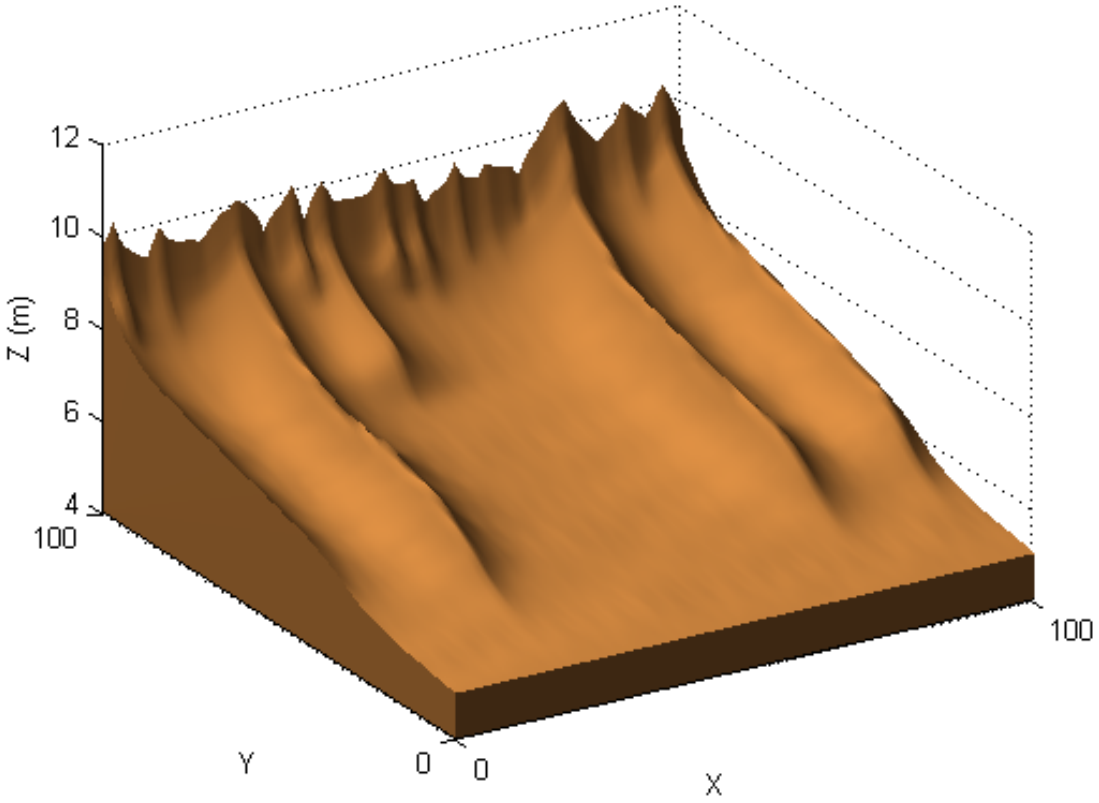


Figure B.7: Small scales added with integrated fractal noise

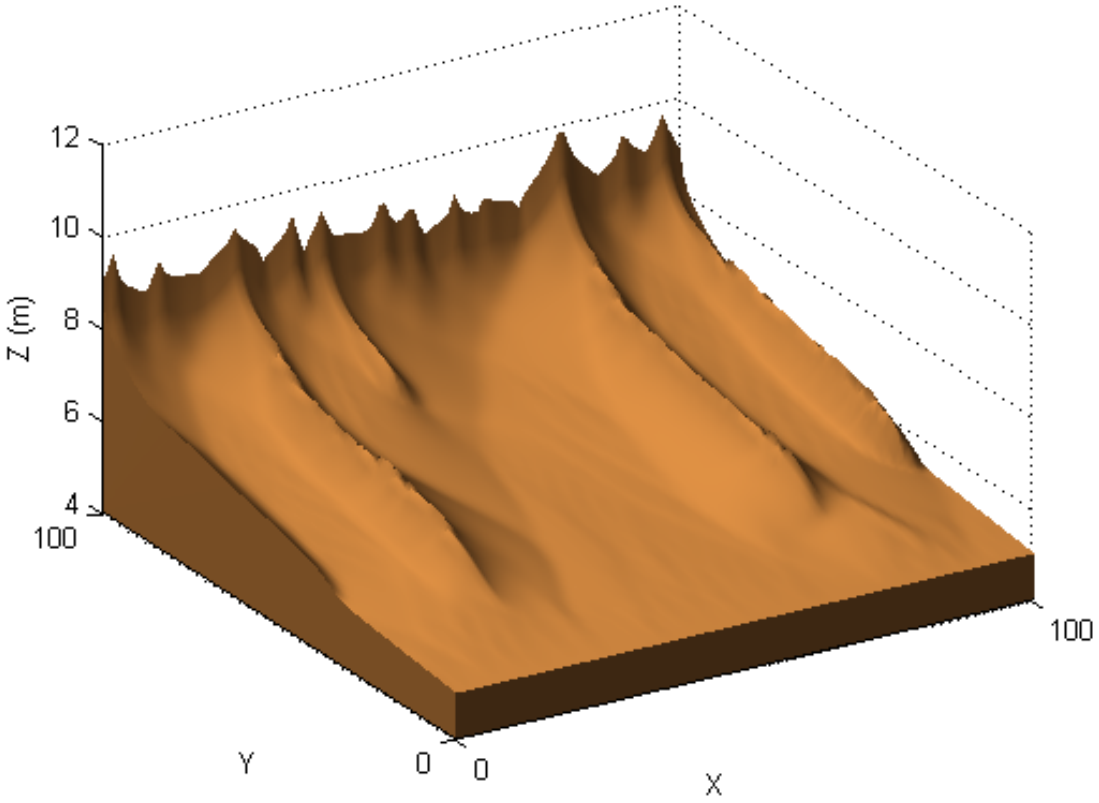


Figure B.8: The basic surface after running in the numerical simulation

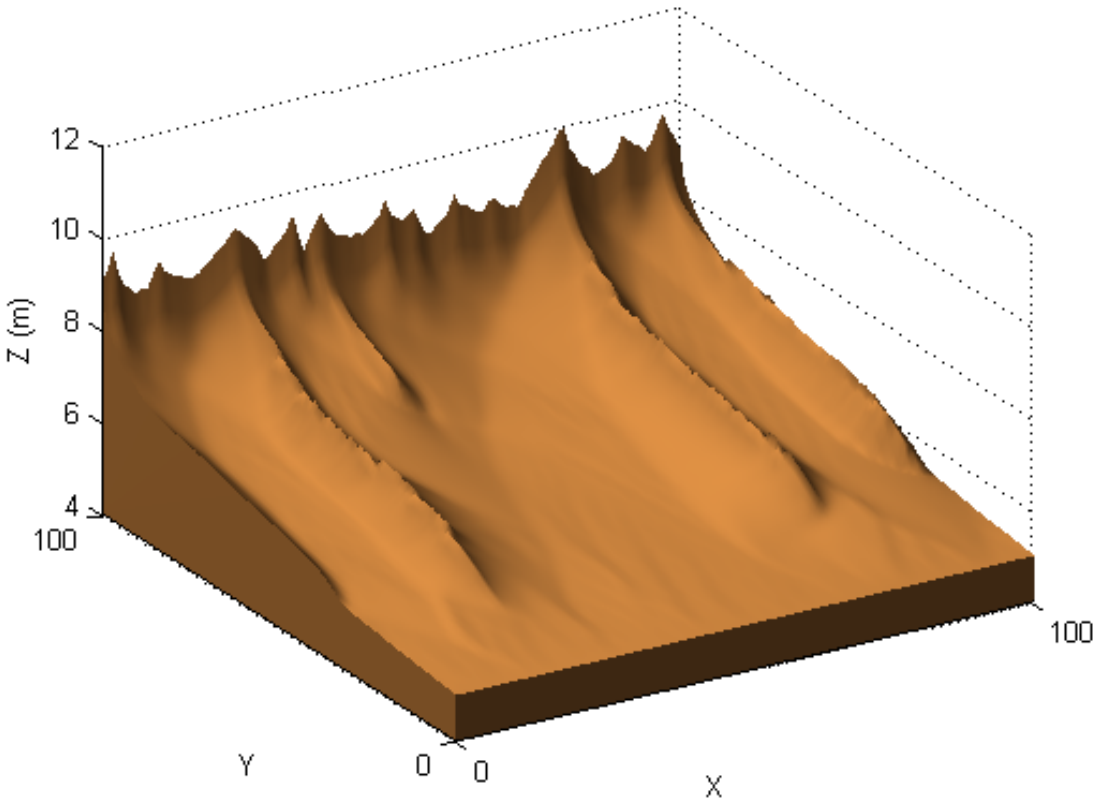


Figure B.9: The piecewise DP interpolated surface after running in the code

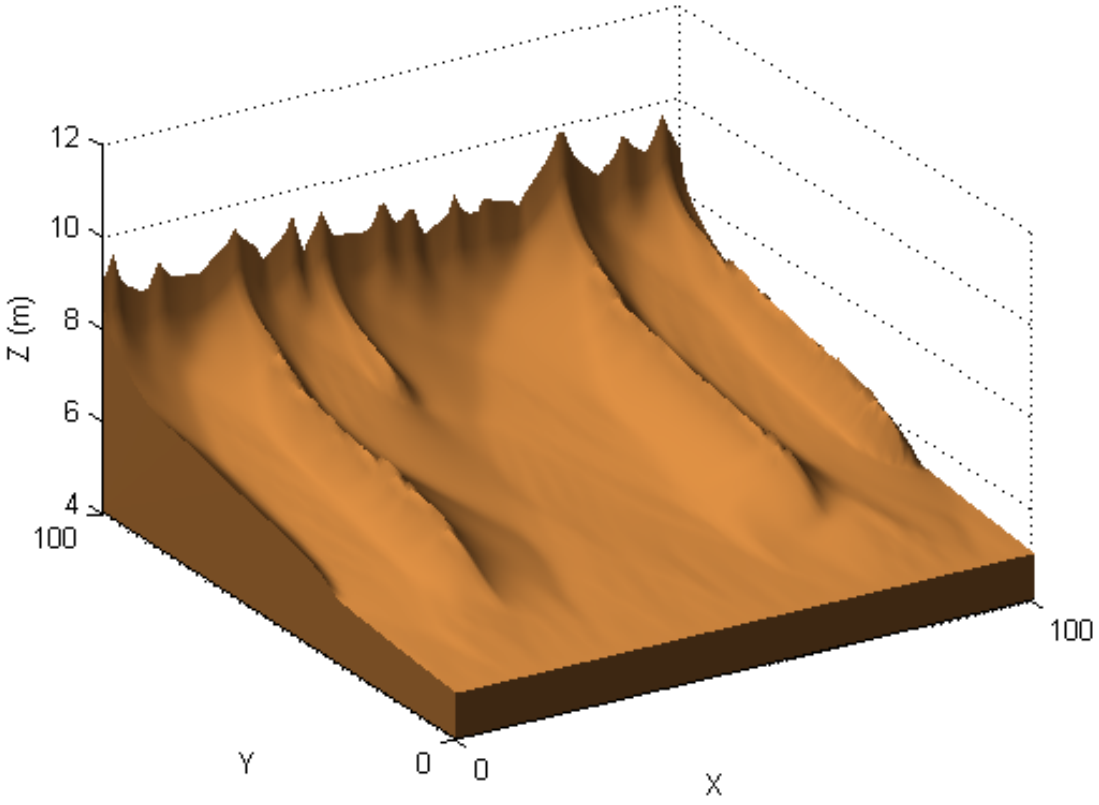


Figure B.10: The independent noise surface after running in the code